IEEE/EIA 12207.2-1997

(A Joint Guide Developed by IEEE and EIA)

IEEE/EIA Guide

Industry Implementation of International Standard ISO/IEC 12207 : 1995

(ISO/IEC 12207) Standard for Information Technology—

Software life cycle processes—Implementation considerations

April 1998

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

ELECTRONIC INDUSTRIES ASSOCIATION ENGINEERING DEPARTMENT

IEEE/EIA 12207.2-1997

(A Joint Guide Developed by IEEE and EIA)

IEEE/EIA Guide

Industry Implementation of International Standard ISO/IEC 12207: 1995

(ISO/IEC 12207) Standard for Information Technology—

Software life cycle processes—Implementation considerations

April 1998

Abstract: ISO/IEC 12207 provides a common framework for developing and managing software. IEEE/EIA 12207.0 consists of the clarifications, additions, and changes accepted by the Institute of Electrical and Electronics Engineers (IEEE) and the Electronic Industries Association (EIA) as formulated by a joint project of the two organizations. IEEE/EIA 12207.2 provides implementation consideration guidance for the normative clauses of IEEE/EIA 12207.0. The guidance is based on software industry experience with the life cycle processes presented in IEEE/EIA 12207.0.
Keywords: acquisition process, audit, configuration management, development process, maintenance process, operation process, quality assurance, supply process, tailoring process, validation, verification

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

Print: ISBN 1-55937-991-X, SH94597
PDF: ISBN 0-7381-0430-2, SS94597

April 1998 *SH94597*

NOTICE

IEEE and EIA Standards and Publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for his particular need. Use of such Standards and Publications is wholly voluntary. Existence of such standards and Publications shall not in any respect preclude any member or nonmember of IEEE and EIA from manufacturing or selling products not conforming to such Standards and Publications, nor shall the existence of such Standards and Publications preclude their voluntary use by those other than IEEE and EIA members, whether the standard is to be used either domestically or internationally.

Standards and publications are approved by IEEE and EIA in accordance with the American National Standards Institute (ANSI) patent policy. By such action, IEEE and EIA do not assume any liability to any patent owner, nor do they assume any obligation whatever to parties adopting the Standard or Publication.

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. IEEE and EIA shall not be responsible for identifying all patents for which a license may be required by an IEEE and EIA standard or for conducting inquiries into the legal validity or scope of those patents that are brought to their attention.

Contents

Annexes

Foreword

1. IEEE/EIA 12207.2 provides guidance in implementing the process requirements of IEEE/EIA 12207.0. The guidance is intended to summarize the best practices of the software industry in the context of the process structure provided by ISO/IEC 12207.

2. While this standard is appropriate for use within the U.S. and globally, it must be understood that some of its provisions are specific to the U.S.; their applicability to international trade situations should be carefully considered.

3. IEEE/EIA 12207.2 is intended for use as a guide to the application of IEEE/EIA 12207.0. To facilitate use, the normative text of IEEE/EIA 12207.0 is included in the guide and enclosed in boxes. The clauses in the boxes are continuous until reaching a clause with guidance. The box is then followed by the appropriate guidance material for the last clause in the box.

4. There are several clauses in section 5 of IEEE/EIA 12207.0 that use the verb “can.” “Can” does not equate to “shall” (a binding provision), “will” (a declaration), “should” (a recommendation), or “may” (a permissible action). As in ISO/IEC 12207, the verbal form “can” is used for statements of possibility and capability, whether material, physical, or causal. “Can” is used in the following clauses: 5.3.6.1, 5.4.4.3, 6.2.3.1, 6.4.2.4 c, 6.4.2.5 c, 6.5.2.3 c, F.2 c of annex B, G.14 and G.15 of annex C, and H.3 of annex D.

5. The preambles for clauses 5, 6, and 7 are informative only.

Introduction

(This introduction is not a part of IEEE/EIA 12207.2-1997, Guide for ISO/IEC 12207, *Standard for information technology—Software life cycle processes—Implementation considerations*.)

At the time this guide was completed, the Joint Industrial Standards Working Group had the following membership:

Leonard L. Tripp, *IEEE Co-chair* **Perry DeWeese**, *EIA Co-chair*

Executive Committee Members

Dennis Ahern
Richard Evans
Lewis Gray
Robert Hegland
Dorothy Kuckuck
David Maibor
James Moore
George Newberry
Adrienne Scott
Raghu Singh
Norma A. Stopyra

Joint Industrial Standards Working Group Members

Chuck Baker
Barbara Bankeroff
Johnny Barrett
Ron Berlack
Barry Boehm
John Bolland
John Bowers
Max Brown
Don Calvert
Quyen Cao
Bruce Capehart
Dennis Carter
Myra M. Chern
John P. Chihorek
Jack Cooper
Raymond Coyle
Paul Croll
Chris Denham
Robert Didrikson
Jim Dobbins
Merlin Dorfman
Cheryl Dorsey
Bernadette Downward
Peter Eirich
Bob Elston
Dennis Faulkner
Marv Gechman
William Gess, Jr.
Marilyn Ginsberg-Finner
John Halase
John Hamlin
Rick Hefner
James H. Heil
Mark Henley
Doug Hewett
John Hoover
Helmut Hummel
Allan Jaworski
Larry Klos
Robert Knickerbocker
Richard Kreke
Jerome Lake
Doug Lange
Amy Laughlin
Milton Lavin
Jonathan Liles
Joan Lovelace
Marv Lubofsky
Ed Martin
Zyggy Martynowkcz
Richard McClellen
Judy McCloskey
Sandy McGill
Fred Mintz
Aretha Moore
Jackie Morman
Gary Motchan
Bart Nigro
George Nowinski
Al Olsen
Myrna L. Olson
Sherry Paquin
Alex Polack
Ken Ptack
Ralph Randall
Paul Reindollar
Walter Richter
Bill Schunk
Keith Shewbridge
Carl A. Singer
Terry Snyder
Reed Sorensen
Don Sova
John Stolzenthaler
Richard F. Storch
Duane Stratton
Robert Tausworthe
Booker T. Thomas
Frederick Tilsworth
Ann Turner
Howard Verne
Ronald L. Wade
David Waxman
Charles Wilson
Grady Wright

The following persons were on the IEEE balloting committee:

Mikhail Auguston
H. Ronald Berlack
William J. Boll
Juris Borzovs
Alan L. Bridges
Kathleen L. Briggs
M. Scott Buck
David W. Burnett
Leslie Chambers
Keith Chan
Antonio M. Cicu
Theo Clarke
François Coallier
Virgil Lee Cooper
W. W. Geoff Cozens
Bostjan K. Derganc
Audrey Dorofee
Carl Einar Dragstedt
Leo Egan
Richard L. Evans
William Eventoff
Jonathan H. Fairclough
John W. Fendrich
Julian Forster
Kirby Fortenberry
Eva Freund
Adel N. Ghannam
Marilyn Ginsberg-Finner
John Garth Glynn
Julio Gonzalez-Sanz
L. M. Gunther
Jon D. Hagar
John Harauz
Robert T. Harley
Herbert Hecht
William Hefley
Manfred Hein
Mark Heinrich
Mark Henley
Umesh P. Hiriyannaiah
John W. Horch
Jerry Huller
Peter L. Hung
Fabrizio Imelio
George Jackelen
Frank V. Jorgensen
Vladan V. Jovanovic
Judith S. Kerner
Robert J. Kierzyk
Dwayne L. Knirk
Thomas M. Kurihara
John B. Lane
J. Dennis Lawrence
Randal Leavitt
Michael Lines
Dieter Look
John Lord
David Maibor
John R. Matras
Tomoo Matsubara
Patrick McCray
Sue McGrath
Jerome W. Mersky
Alan Miller
James W. Moore
Pavol Navrat
Myrna L. Olson
Mike Ottewill
Gerald L. Ourada
Lalit M. Patnaik
Mark Paulk
John G. Phippen
Peter T. Poon
Lawrence S. Przybylski
Ann Reedy
Annette D. Reilly
Dennis Rilling
Andrew P. Sage
Helmut Sandmayr
Hans Schaefer
Robert W. Shillato
Katsutoshi Shintani
Carl A. Singer
Raghu P. Singh
Alfred R. Sorkowitz
Donald W. Sova
Luca Spotorno
Julia Stesney
Norma Stopyra
Fred J. Strauss
John Swearingen
Toru Takeshita
Douglas H. Thiele
Booker Thomas
Patricia Trellue
Theodore J. Urbanowicz
Glenn D. Venables
Charles J. Wertz
Scott A. Whitmire
P. A. Wolfgang
Paul R. Work
Weider D. Yu
Janusz Zalewski
Zhi Ying Zhou
Geraldine Zimmerman
Peter F. Zoll

When the IEEE Standards Board approved this standard on 9 December 1997, it had the following membership:

Donald C. Loughry, *Chair* **Richard J. Holleman,** *Vice Chair*
Andrew G. Salem, *Secretary*

Clyde R. Camp
Stephen L. Diamond
Harold E. Epstein
Donald C. Fleckenstein
Jay Forster*
Thomas F. Garrity
Donald N. Heirman
Jim Isaak
Ben C. Johnson
Lowell Johnson
Robert Kennelly
E. G. "Al" Kiener
Joseph L. Koepfinger*
Stephen R. Lambert
Lawrence V. McCall
L. Bruce McClung
Marco W. Migliaro
Louis-François Pau
Gerald H. Peterson
John W. Pope
Jose R. Ramos
Ronald H. Reimer
Ingo Rüsch
John S. Ryan
Chee Kiow Tan
Howard L. Wolfman

*Member Emeritus

Also included are the following nonvoting IEEE Standards Board liaisons:

Satish K. Aggarwal
Alan H. Cookson

Information technology—Software life cycle processes—Implementation considerations

1 Scope

This guide provides implementation consideration guidance for the normative clauses of IEEE/EIA 12207.0. The guidance is based on software industry experience with the life cycle processes presented in IEEE/EIA 12207.0.

1.1 Limitations

In this guide the use of “organizational process” denotes all of the processes as adapted by an organization, whereas the use of “organizational life cycle process” refers to the process(es) in clause 7. A thorough understanding of the organizational life cycle processes in clause 7 will help the user of IEEE/EIA 12207.0 understand what should be in place prior to performing the acquisition, supply, development, maintenance, and/or operation process(es).

1.2 Tailoring

The term “tailoring” is used frequently in the text quoted from ISO/IEC 12207. The concept of unconstrained tailoring is discouraged in IEEE/EIA 12207.0. In general, the term tailoring should be construed as meaning simply that tasks outside the scope of work of the relevant contract may be omitted. Additional guidance is provided in annex B of this guide.

2 Normative references

The following standard contains provisions which, through reference in this text, constitute provisions of this guide. At the time of publication, the edition indicated was valid. All standards are subject to revision, and parties to agreements based on this guide are encouraged to investigate the possibility of applying the most recent editions of the standard indicated below. When the following standard is superseded by an approved revision, the revision shall apply.

IEEE/EIA 12207.0-1996, *Information technology—Software life cycle processes*.

Refer to clause 2 of IEEE/EIA 12207.0 for normative references associated with IEEE/EIA 12207.0.

3 Definitions

GUIDANCE:

1—The definition of software unit in IEEE/EIA 12207.0 (3.28) is identical to that of ISO/IEC 12207. This definition, however, may represent one view of software unit. There may not be a common agreement within the software community on “software unit.” The simplest notion of software unit is that it is what a project considers the smallest part of its software product. For this guide, the interpretation of the term “software unit” is the following: (a) “a separately compilable piece of code,” (b) any other appropriate definition, or (c) replaced with other more useful terms and concepts for the project.

2—The term “compilable” refers to compilation, assembly, interpreting, translation, etc. The intent is for the computer to “translate” a unit written in a higher-order language to its machine instructions so that it can execute them. The computer can “compile” a software unit separately from other software units. However, compilation of a software unit with a test generator, for example, is not considered separate compilation. A software unit is the lowest element in one or more software components. A software component is not necessarily immediately partitioned into software units. A software component is refined into levels containing software units that can be coded, compiled, and tested.

4 Application

This guide is an implementation guide to IEEE/EIA 12207.0.

5 Primary life cycle processes

This clause defines the following primary life cycle processes:

1) Acquisition process;
2) Supply process;
3) Development process;
4) Operation process;
5) Maintenance process.

The activities and tasks in a primary process are the responsibility of the organization initiating and performing that process. This organization ensures that the process is in existence and functional.

GUIDANCE:

1—IEEE/EIA 12207.0 supports the use of integrated product teams (i.e., interdependent multidisciplinary teams, such as acquirer-supplier/developer, contractor-subcontractor, etc.) as one possible means to accomplish goals, identify and resolve issues, and make sound and timely recommendations to facilitate decision making.

2—IEEE/EIA 12207.0 uses "requirement" in many contextual ways. When a qualifier is provided (e.g., acquisition, system, software, test, contract), the qualifier implies the source(s) and receiver(s) of that requirement. When requirement is used without a qualifier, the meaning of that requirement is local to the specific activity or task.

3—Annexes A through E of this guide contain the normative annexes from IEEE/EIA 12207.0 and provide additional guidance on tailoring, additional guidance on compliance, additional guidance on life cycle processes objectives, additional guidance on life cycle data objectives, and ISO/IEC 12207 errata. Informative annexes F through M of this guide provide guidance on use of reusable software products, candidate joint management reviews, software measurement categories, guidance on development strategies and build planning, suggested category and priority classifications for problem reporting, guidance on software product evaluations, guidance on risk management, and life cycle processes references.

4—The activities within each of the primary life cycle processes may be performed iteratively and in any order deemed appropriate to produce the required software product or service. See 1.5 of IEEE/EIA 12207.0, third paragraph, for additional information.

5—Figure 1 shows the relationships between the life cycle processes of IEEE/EIA 12207.0 and roles.

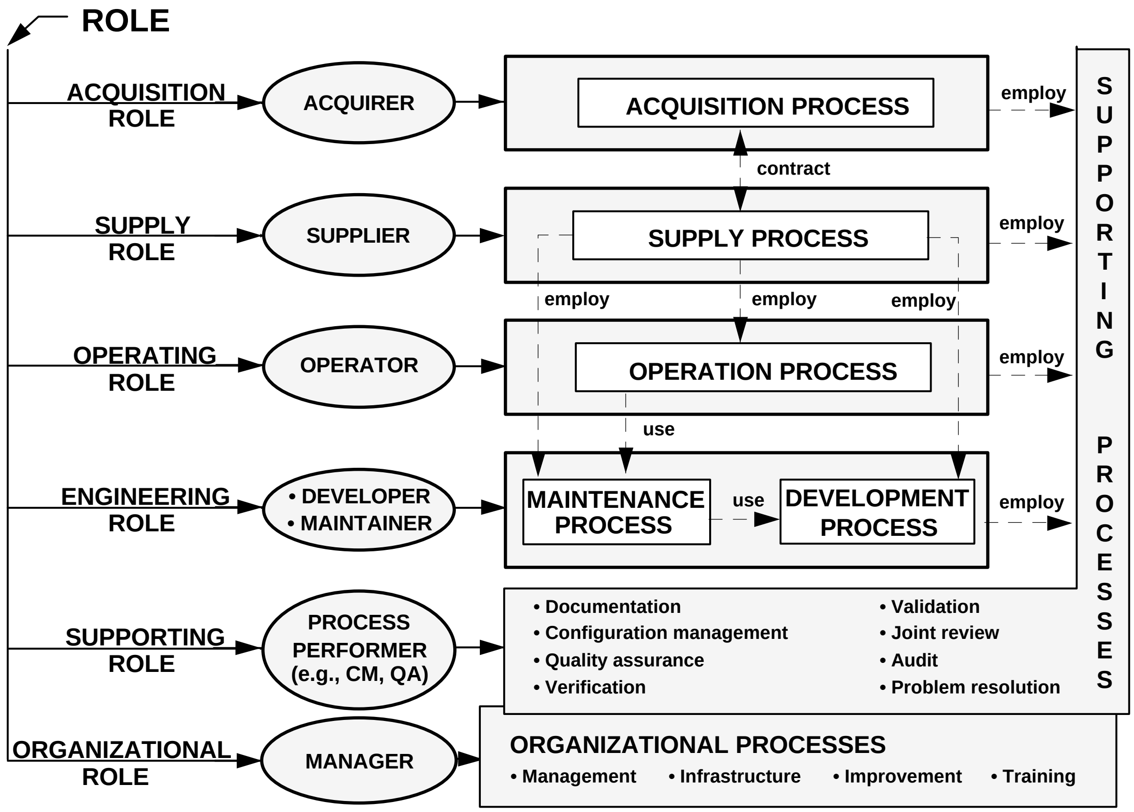

Figure 1—Relationship of primary life cycle processes and roles

5.1 Acquisition process

The Acquisition Process contains the activities and tasks of the acquirer. The process begins with the definition of the need to acquire a system, software product or software service. This process continues with the preparation and issue of a request for proposal, selection of a supplier, and management of the acquisition process through to the acceptance of the system, software product or software service.

The individual organization having the need may be called the owner. The owner may contract any or all of the acquisition activities to an agent who will in turn conduct those activities according to the Acquisition Process. The acquirer in this subclause may be the owner or the agent.

The acquirer manages the Acquisition Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4).

List of activities: This process consists of the following activities:

1) Initiation;
2) Request-for-Proposal [-tender] preparation;
3) Contract preparation and update;
4) Supplier monitoring;
5) Acceptance and completion.

GUIDANCE:

1—Preferably, a project's implementation of IEEE/EIA 12207.0 should be specified with respect to the organization's claim of compliance to IEEE/EIA 12207.0. See annex B of this guide for further guidance on compliance.

2—An acquirer performing the acquisition process is conducting an "acquisition project." As part of the execution of the project, the acquirer should apply the organizational acquisition process. Nothing in this clause is intended to require that the acquirer should attempt to levy any specific process requirements upon the supplier, developer, operator, or maintainer beyond that of compliance with this standard.

3—IEEE/EIA 12207.0 uses "requirement" in many contextual ways. When a qualifier is provided (e.g., acquisition, system, software, test, contract), the qualifier implies the source(s) and receiver(s) of that requirement. When requirement is used without a qualifier, the meaning of that requirement is local to the specific activity or task.

4—The acquirer should assess the capability and qualifications of a potential supplier's processes both at an organizational level (if implemented) and as adapted for the specific project.

5—IEEE Std 1062[1] provides additional guidance regarding the acquisition process, including supplier selection.

6—Figure 2 represents the acquisition process.

[1] Life cycle process references can be found in annex M.

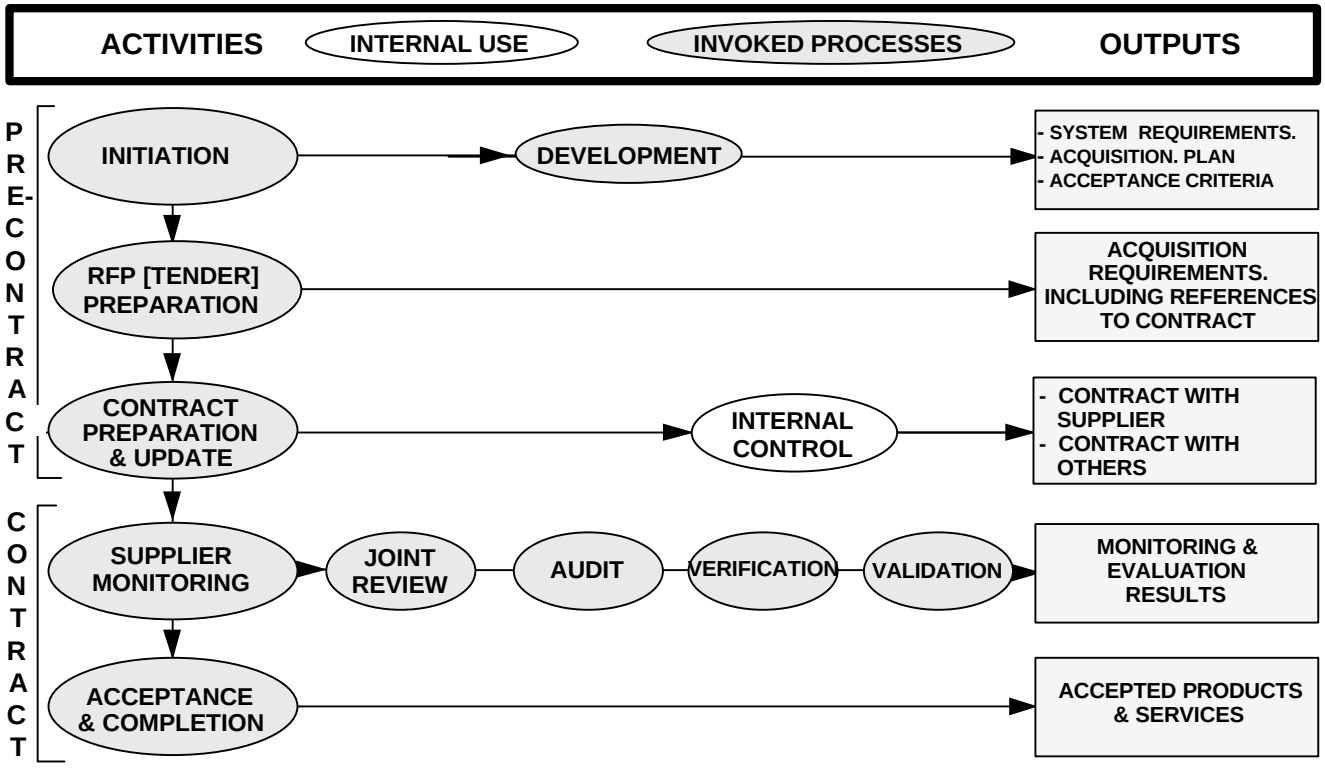

Figure 2—Acquisition process

5.1.1 Initiation. This activity consists of the following tasks:

5.1.1.1 The acquirer begins the acquisition process by describing a concept or a need to acquire, develop, or enhance a system, software product or software service.

GUIDANCE:

1—All disciplines affected by the system, software product, or software service to be acquired should be involved in concept definition. The operational concept should be modified/updated periodically to reflect current needs. The operational concept should address the full life cycle (acquisition, supply, development, operation, maintenance) of the system, software product, or software service. The description of the operational concept may be used to (1) obtain consensus among the acquirer, supplier, developer, maintenance, and user agencies on the operational and maintenance concept for the proposed system and (2) help users understand and adapt to operational and maintenance changes needed as a result of modifications to a currently operating system undergoing reengineering, addition of new capabilities, or other modifications.

2—Interpret “begins” as “will begin” to meet the intent of 5.1.1.1.

5.1.1.2 The acquirer will define and analyze the system requirements. The system requirements should include business, organizational and user as well as safety, security, and other criticality requirements along with related design, testing, and compliance standards and procedures.

GUIDANCE:

1—The system requirements should be written at an appropriate level, based on the level of knowledge about the operational concept and the system to be acquired. The system requirements evolve as knowledge is gained by all parties involved. Many methods are available for use in defining system requirements (e.g., concept exploration, prototyping).

2—Definition of system safety and security requirements should also include what the system must not do.

3—In systems containing software with safety requirements, it may be appropriate to apply IEEE Std 1228. RTCA DO-178B may also be a useful reference.

4—In systems containing software with high reliability requirements, it may be appropriate to apply IEEE Std 982.1 to formulate quantifiable reliability requirements.

5.1.1.3 If the acquirer retains a supplier to perform system requirements analysis, the acquirer will approve the analyzed requirements.

GUIDANCE:

The acquirer should ensure the resulting system requirements provide appropriate flexibility for designing, modifying, or expanding the delivered system. For example, in appropriate situations, system requirements may include an “open systems” approach. Alternately, system requirements may call for use of, or consistency with, a domain specific architecture or the architecture of other software systems in the operating environment.

5.1.1.4 The acquirer may perform the definition and analysis of software requirements by itself or may retain a supplier to perform this task.

5.1.1.5 The Development Process (5.3) should be used to perform the tasks in 5.1.1.2 and 5.1.1.4.

GUIDANCE:

Specifically, 5.3.1 through 5.3.4.3 of IEEE/EIA 12207.0 should be used for performance of system requirements analysis and software requirements analysis.

5.1.1.6 The acquirer will consider options for acquisition against analysis of appropriate criteria to include risk, cost and benefits for each option. Options include:

a) Purchase an off-the-shelf software product that satisfies the requirements.
b) Develop the software product or obtain the software service internally.
c) Develop the software product or obtain the software service through contract.
d) A combination of a, b, and c above.
e) Enhance an existing software product or service.

GUIDANCE:

The acquirer should solicit inputs from potential suppliers when considering the options of 5.1.1.6.

5.1.1.7 When an off-the-shelf software product is to be acquired, the acquirer will ensure the following conditions are satisfied:

a) The requirements for the software product are satisfied.
b) The documentation is available.
c) Proprietary, usage, ownership, warranty and licensing rights are satisfied.
d) Future support for the software product is planned.

5.1.1.8 The acquirer should prepare, document and execute an acquisition plan. The plan should contain the following:

a) Requirements for the system;
b) Planned employment of the system;
c) Type of contract to be employed;
d) Responsibilities of the organizations involved;
e) Support concept to be used;
f) Risks considered as well as methods to manage the risks.

GUIDANCE:

1—The acquisition plan should define the project acquisition process and should describe the relationship of the project's acquisition process to the organizational acquisition process.

2—For 5.1.1.8 d), the supplier is responsible for coordinating the contract with subcontractor(s). If the project uses multifunctional teams, the acquirer and subcontractor(s) may also be team participants.

3—The acquirer should establish and document in the acquisition plan a software measurement program. The program should be used to effectively plan and track software activities and tasks across the life cycle; aid in managing cost, schedule, and technical risks; and integrate software measurement with existing program management and software programming disciplines established for the project.

5.1.1.9 The acquirer should define and document the acceptance strategy and conditions (criteria).

GUIDANCE:

Since acceptance of the product or service usually relieves the supplier of contractual responsibilities, the acquirer should carefully consider the acceptance strategy and conditions. Factors to consider include (1) the extent, requirements, and location of developer-performed qualification testing; (2) support for acquirer-performed operational testing; (3) establishment of an appropriate maintenance concept, including spares, support equipment, etc.; (4) interim supplier/developer support; (5) successful software installation and transition to a maintenance activity; (6) training; and (7) warranties. The period of contractual responsibility, as well as obligations of the supplier following acquirer acceptance, should be defined.

5.1.2 Request-for-proposal [-tender] preparation. This activity consists of the following tasks:

5.1.2.1 The acquirer should document the acquisition requirements (e.g., request for proposal), the content of which depends upon the acquisition option selected in 5.1.1.6. The acquisition documentation should include, as appropriate:

a) System requirements;
b) Scope statement;
c) Instructions for bidders;
d) List of software products;
e) Terms and conditions;
f) Control of subcontracts;
g) Technical constraints (e.g., target environment).

GUIDANCE:

1—The acquirer should specify the level of acquirer participation throughout the supplier's project execution to provide the required software product or service.

2—The acquirer may request, in the request for proposal (RFP), access to project plans and planning data, proposed processes, standards, and practices. Access may be requested for (1) evaluation and reference during supplier selection and during development/maintenance/operation and (2) support after transition, if necessary. When appropriate to project characteristics, acquirer review of plans and other specifically identified procedures, practices, and work products should occur during source selection and contract negotiations. In

some cases, review may take place after contract award. The identification of items for review may be done using table 1 of IEEE/EIA 12207.1.

3—The acquirer may specify in the RFP use of IEEE/EIA 12207.0 and other criteria as the basis for evaluation of supplier processes and plan(s) and make the evaluation criteria known. When more than one standard is specified as a basis, a clear order of precedence should be established in order to prevent conflicting interpretations.

4—The acquirer should use the draft RFP process to elicit inputs from suppliers to refine requirements and contractual considerations.

5—In the RFP, the acquirer should request the description of a framework for capturing software measures. The software measurement framework should define a flexible measurement process that directly supports the software life cycle process and related tasks and activities as implemented for the project. The measurement process should be based on the selection and analysis of appropriate software measures to address the specific project software issues, constraints, and objectives as determined by the acquirer and supplier. Software measurement data should be made available, preferably by automated data access methods, to the acquirer and supplier as part of the acquirer-supplier agreement.

6—The acquirer should include in the RFP the definition of the need for independent validation and verification, joint reviews, audits, and the process to be used to flow information to the acquirer from the operations/maintenance/development processes.

5.1.2.2 The acquirer should determine which processes, activities, and tasks of the International Standard are appropriate for the project and should tailor them accordingly. Especially, the acquirer should specify the applicable supporting processes (clause 6) and their performing organizations, including responsibilities (if other than supplier), so that the suppliers may, in their proposals, define the approach to each of the specified supporting processes. The acquirer will define the scope of those tasks that reference the contract.

GUIDANCE:

1—Performing organization, as intended in this clause, refers to the acquirer or supplier organization, not the functional organization (e.g., configuration management, quality assurance) within either.

2—The acquirer should review the established documentation standards and the proposed project-specific application of the supplier and determine if the proposed documentation will meet the acquirer's needs. If the acquirer needs additional or different documentation than that offered in the supplier's proposal, all affected parties should come to an agreement regarding the documentation during contract negotiation or establish a framework for post-award agreement.

3—The purpose of the acquirer's tailoring of the standard is to specify the scope of activities to be covered by the contract. The process framework of the standard can be very helpful in achieving effective communication between acquirer and supplier regarding the activities to be performed in the anticipated work.

4—When appropriate to project characteristics, acquirer review of plans and other specifically identified procedures, practices, and work products should occur during source selection and contract negotiations. In some cases, further refinement and review may take place after contract award. The identification of items for review may be done using table 1 of IEEE/EIA 12207.1.

5—The acquirer should only acquire life cycle data that it expects to use during the contractual period or afterward. Information items described in 4.3 of IEEE/EIA 12207.1 should be tailored to their component level to avoid developing and acquiring useless data. Deliverable data should have content, location, format, media, and packaging that is appropriate for the organization or project where they are developed and recorded. The acquirer should ensure the delivered data meet the intended needs.

5.1.2.3 The acquisition documentation will also define the contract milestones at which the supplier's progress will be reviewed and audited as part of monitoring the acquisition (see 6.6 and 6.7).

GUIDANCE:

Software review schedules should support system-level and milestone reviews. Contract milestones should be based on clearly defined entry and exit criteria.

5.1.2.4 The acquisition requirements should be given to the organization selected for performing the acquisition activities.

5.1.3 Contract preparation and update. This activity consists of the following tasks:

5.1.3.1 The acquirer should establish a procedure for supplier selection including proposal evaluation criteria and requirements compliance weighting.

GUIDANCE:

Evaluation criteria may include compliance with national standards, including IEEE/EIA 12207.0, and process maturity.

5.1.3.2 The acquirer should select a supplier based upon the evaluation of the suppliers' proposals, capabilities, and other factors that need to be considered.

GUIDANCE:

Other factors for consideration include capability/maturity, domain expertise, past performance, financial stability, commitment to continuous improvement, location, and staffing.

5.1.3.3 The acquirer may involve other parties, including potential suppliers, before contract award, in tailoring this International Standard for the project. However, the acquirer will make the final decision on the tailoring. The acquirer will include or reference the tailored International Standard in the contract.

GUIDANCE:

In some cases, the requirements in 5.1.3.3 may be inappropriate or difficult to implement. The following is provided as a means of achieving the intent of the clause. Acquirer tailoring should be applied only to define the scope of work to be acquired. Rather than develop a unique tailoring for a project, the acquirer should solicit proposals based upon suppliers' organizational processes and evaluate the suppliers' proposed processes. Additional guidance on tailoring and compliance may be found in annexes A and B, respectively, of this guide.

5.1.3.4 The acquirer will then prepare and negotiate a contract with the supplier, that addresses the acquisition requirements, including the cost and schedule, of the software product or service to be delivered. The contract will address proprietary, usage, ownership, warranty and licensing rights associated with the reusable off-the-shelf software products.

GUIDANCE:

1—The acquirer should identify for the supplier the retention period for records, either directly or indirectly by reference to other legal requirements.

2—Acquisition requirements should also address the trade-offs between cost, schedule, performance, business objectives, safety, product quality level, etc.

3—The acquirer should establish with the supplier a means for controlling and making changes to both contractual requirements and requirements of the delivered system or service.

4—The contract should address proprietary, usage, ownership, warranty, and licensing rights associated with all the software products to be acquired.

5—The acquirer should define requirements and preferences and negotiate with the supplier to define (1) what information should be delivered to the acquirer, (2) what information should be accessible to the acquirer,

(3) what information requires approval from the acquirer when first created or subsequently modified, (4) any format and content requirements for the information, and (5) when information should be produced.

5.1.3.5 Once the contract is underway, the acquirer will control changes to the contract through negotiation with the supplier as part of a change control mechanism. Changes to the contract will be investigated for impact on project plans, costs, benefits, quality, and schedule.

NOTE—The acquirer determines whether the term "contract" or "agreement" is to be used in the application of this International Standard.

GUIDANCE:

1—The investigation of the impact of changes to the contract should include all potential significant risks.

2—Changes to the contract should not be used as a method to force changes in a supplier's organizational processes.

5.1.4 Supplier monitoring. This activity consists of the following tasks:

5.1.4.1 The acquirer will monitor the supplier's activities in accordance with the Joint Review Process (6.6) and the Audit Process (6.7). The acquirer should supplement the monitoring with the Verification Process (6.4) and the Validation Process (6.5) as needed.

GUIDANCE:

1—The acquirer should have the opportunity to examine the supplier's/developer's/maintainer's/operator's processes for developing, modifying, or otherwise supporting the system/software to ensure cost-effective expenditure of resources and a quality product that meets user needs within project cost and schedule constraints.

2—The acquirer and supplier should both participate in joint reviews. The requirements/responsibilities of all parties involved in joint management and joint technical reviews should be documented. Annex G of this guide contains candidate joint management reviews.

3—One or more methods may be used to monitor supplier's activities. These methods include software measurements, integrated product teams, on-site or remote access to digital data, and joint reviews.

4—The acquirer may use data from the supplier's software measurement program as input to its own software measurement program for supplier monitoring.

5.1.4.2 The acquirer will cooperate with the supplier to provide all necessary information in a timely manner and resolve all pending items.

GUIDANCE:

Arrangements should be established to ensure both the acquirer and the supplier cooperate to provide necessary information and work together to resolve problems and risks.

5.1.5 Acceptance and completion. This activity consists of the following tasks:

5.1.5.1 The acquirer should prepare for acceptance based on the defined acceptance strategy and criteria. The preparation of test cases, test data, test procedures, and test environment should be included. The extent of supplier involvement should be defined.

GUIDANCE:

1—The acquirer should consider including the user in preparation for acceptance testing. Depending on the life cycle support plans for the system/software, the acquirer should also include operations and maintenance organizations in preparation for acceptance testing.

2—The results of other processes (e.g., qualification test, verification, validation) may be used as part of acceptance and completion.

5.1.5.2 The acquirer will conduct acceptance review and acceptance testing of the deliverable software product or service and will accept it from the supplier when all acceptance conditions are satisfied. The acceptance procedure should comply with the provisions of 5.1.1.9.

GUIDANCE:

1—Depending on the nature of the product developed, the acquirer may accept only the software product, the software product and associated computer hardware, or the entire system, which includes embedded software.

2—As specified in 5.1.5, the acquirer determines the acceptance strategy. The acquirer may elect to perform separate acceptance testing and review with developer support, task the developer to perform all testing and base acceptance upon adequate review of the developer's test results, or some variation.

5.1.5.3 After acceptance, the acquirer should take the responsibility for the configuration management of the delivered software product (see 6.2).

NOTE—The acquirer may install the software product or perform the software service in accordance with instructions defined by the supplier.

No further guidance provided.

5.2 Supply process

The Supply Process contains the activities and tasks of the supplier. The process may be initiated either by a decision to prepare a proposal to answer an acquirer's request for proposal or by signing and entering into a contract with the acquirer to provide the system, software product or software service. The process continues with the determination of procedures and resources needed to manage and assure the project, including development of project plans and execution of the plans through delivery of the system, software product or software service to the acquirer.

The supplier manages the Supply Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4).

List of activities: This process consists of the following activities:

1) Initiation;
2) Preparation of response;
3) Contract;
4) Planning;
5) Execution and control;
6) Review and evaluation;
7) Delivery and completion.

GUIDANCE:

1—IEEE/EIA 12207.0 uses "requirement" in many contextual ways. When a qualifier is provided (e.g., acquisition, system, software, test, contract), the qualifier implies the source(s) and receiver(s) of that requirement. When requirement is used without a qualifier, the meaning of that requirement is local to the specific activity or task.

2—Figure 3 represents the supply process.

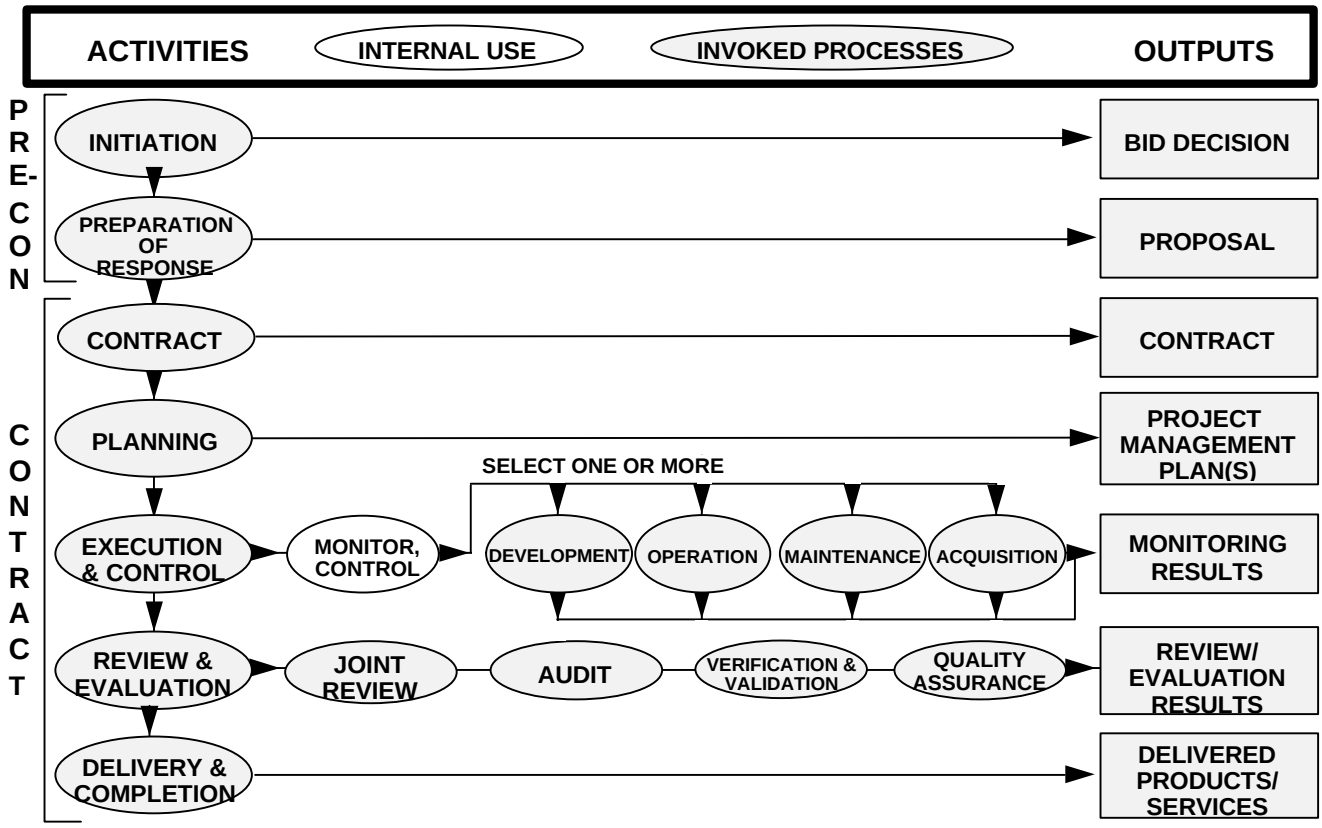

Figure 3—Supply process

5.2.1 Initiation. This activity consists of the following tasks:

5.2.1.1 The supplier conducts a review of requirements in the request for proposal taking into account organizational policies and other regulations.

GUIDANCE:

Preferably, a project's implementation of IEEE/EIA 12207.0 should be specified with respect to the organization's claim of compliance to IEEE/EIA 12207.0. See annex B of this guide for further guidance on compliance.

5.2.1.2 The supplier should make a decision to bid or accept the contract.

5.2.2 Preparation of response. This activity consists of the following task:

5.2.2.1 The supplier should define and prepare a proposal in response to the request for proposal, including its recommended tailoring of this International Standard.

GUIDANCE:

1—The supplier should obtain inputs from the developer, operator, maintainer, selected support, and organizational entities, as applicable, to assist in proposal preparation.

2—The proposal is a means for the supplier to propose the defined processes of the supplier's organization or to recommend contract considerations (e.g., contract type, life cycle considerations, milestones).

5.2.3 Contract. This activity consists of the following tasks:

5.2.3.1 The supplier shall negotiate and enter into a contract with the acquirer organization to provide the software product or service.

5.2.3.2 The supplier may request modification(s) to the contract as part of the change control mechanism.

5.2.4 Planning. This activity consists of the following tasks:

5.2.4.1 The supplier shall conduct a review of the acquisition requirements to define the framework for managing and assuring the project and for assuring the quality of the deliverable software product or service.

GUIDANCE:

The project management framework should be in the context of the organization's defined processes as adapted for the project.

5.2.4.2 If not stipulated in the contract, the supplier shall define or select a software life cycle model appropriate to the scope, magnitude, and complexity of the project. The processes, activities, and tasks of this International Standard shall be selected and mapped onto the life cycle model.

GUIDANCE:

1—The software life cycle model selected should be jointly agreed upon by the acquirer and the supplier. The software life cycle model(s) should be constructed with the activities and tasks of the supply, development, operation, maintenance, and applicable supporting and organizational processes, even if only one of the aforementioned primary processes would be executed.

2—Typically, a project premise is that the organization's selected and approved life cycle models exist and are available for use by the projects. The organization, using clause 7 of IEEE/EIA 12207.0, has determined and documented its business requirements and has determined the life cycle model for use to support the business. A project then selects which life cycle model to use based on the need of its project(s). The project, based on its project plan, will map the activities and tasks to the life cycle. The mapping of the activities and tasks of the Primary Processes to the selected life cycle model should be accompanied by the relevant Supporting Processes. This mapping is required for each iteration or pass through the incremental and evolutionary models. Other project activities should also be mapped to the model.

3—See annex I of this guide for further guidance on determining the model types.

5.2.4.3 The supplier shall establish requirements for the plans for managing and assuring the project and for assuring the quality of the deliverable software product or service. Requirements for the plans should include resource needs and acquirer involvement.

GUIDANCE:

1—The supplier should establish the framework for capturing software measures consistent with the acquirer's guidance in the RFP. This measurement framework should directly support the project software life cycle process and related tasks and activities. It should provide an overall structure for identifying and prioritizing software issues, selecting appropriate measures to address the identified issues, integrating the measurement requirements into the engineering and management processes, analyzing and reporting the measurement results, and taking appropriate action. The measurement framework should be implemented to support supplier and acquirer information throughout the project life cycle and should address the analysis of the feasibility of the project plans and the tracking of project performance against the plans.

2—Supplier's plans should be developed in accordance with IEEE/EIA 12207.1.

5.2.4.4 Once the planning requirements are established, the supplier shall consider the options for developing the software product or providing the software service, against an analysis of risks associated with each option. Options include:

a) Develop the software product or provide the software service using internal resources.
b) Develop the software product or provide the software service by subcontracting.
c) Obtain off-the-shelf software products from internal or external sources.
d) A combination of a, b, and c above.

5.2.4.5 The supplier shall develop and document project management plan(s) based upon the planning requirements and options selected in 5.2.4.4. Items to be considered in the plan include but are not limited to the following:

a) Project organizational structure and authority and responsibility of each organizational unit, including external organizations;

b) Engineering environment (for development, operation, or maintenance, as applicable), including test environment, library, equipment, facilities, standards, procedures, and tools;

c) Work breakdown structure of the life cycle processes and activities, including the software products, software services and non-deliverable items, to be performed together with budgets, staffing, physical resources, software size, and schedules associated with the tasks;

d) Management of the quality characteristics of the software products or services. Separate plans for quality may be developed.

e) Management of the safety, security, and other critical requirements of the software products or services. Separate plans for safety and security may be developed.

f) Subcontractor management, including subcontractor selection and involvement between the subcontractor and the acquirer, if any;

g) Quality assurance (see 6.3);

h) Verification (see 6.4) and validation (see 6.5); including the approach for interfacing with the verification and validation agent, if specified;

i) Acquirer involvement; that is, by such means as joint reviews (see 6.6), audits (see 6.7), informal meetings, reporting, modification and change; implementation, approval, acceptance, and access to facilities;

j) User involvement; by such means as requirements setting exercises, prototype demonstrations and evaluations;

k) Risk management; that is management of the areas of the project that involve potential technical, cost, and schedule risks;

l) Security policy; that is, the rules for need-to-know and access-to-information at each project organization level;

m) Approval required by such means as regulations, required certifications, proprietary, usage, ownership, warranty and licensing rights;

n) Means for scheduling, tracking, and reporting;

o) Training of personnel (see 7.4).

GUIDANCE:

1—In developing and documenting project management plan(s), the supplier should obtain support from the developer, operator, maintainer, selected support and organizational entities, as applicable, based on organizational processes conforming to IEEE/EIA 12207.0.

2—For 5.2.4.5 i) and 5.2.4.5 j), the supplier should list the areas and levels of acquirer participation in the supply process as specified in the contract (e.g., review of project plans, testing). This information should be added to the appropriate sections of the project management plan.

3—For 5.2.4.5 k), the annex on risk management, annex L of this guide, should be referenced.

4—For 5.2.4.5 n), the software measurement framework established in 5.2.4.3 should be used to capture data.

5—The supplier's processes for the project should be specified with respect to the supplier's organizational processes and policies.

6—The supplier should record the transition strategy and approach in the project management plan(s).

5.2.5 Execution and control. This activity consists of the following tasks:

5.2.5.1 The supplier shall implement and execute the project management plan(s) developed in 5.2.4.

GUIDANCE:

1—The project management plan(s) should be updated/modified, as needed, to reflect changes in requirements, policies, and procedures.

2—The acquirer should be kept informed of updates to the plan(s).

5.2.5.2 The supplier shall:

a) Develop the software product in accordance with Development Process (5.3).
b) Operate the software product in accordance with Operation Process (5.4).
c) Maintain the software product in accordance with Maintenance Process (5.5).

GUIDANCE:

Each of these processes are implementations of the respective organization's processes.

5.2.5.3 The supplier shall monitor and control the progress and the quality of the software products or services of the project throughout the contracted life cycle. This shall be an ongoing, iterative task, which shall provide for:

a) Monitoring progress of technical performance, costs, and schedules and reporting of project status;

b) Problem identification, recording, analysis, and resolution.

GUIDANCE:

1—Risk management should be included in the ongoing activities listed in 5.2.5.3 a).

2—The supplier should use the problem resolution process (6.8) for the items in 5.2.5.3 b).

3—The software measurement framework established in 5.2.4.3 should be used for capturing data.

5.2.5.4 The supplier shall manage and control the subcontractors in accordance with the Acquisition Process (5.1). The supplier shall pass down all contractual requirements necessary to ensure that the software product or service delivered to the acquirer is developed or performed in accordance with the prime-contract requirements.

GUIDANCE:

1—The subcontract should define only those tasks the subcontractor is to perform and flow down only those requirements from the prime contract that are applicable to the work the subcontractor is to perform.

2—When the supplier passes requirements down to subcontractor(s), the supplying contractor assumes the role of an acquirer, and the subcontractor assumes the role of a supplier. The resulting acquirer-supplier relationship is distinct from the existing higher-level acquirer-supplier relationship. This illustrates the principle that "acquirer" and "supplier" refer to roles that may be assumed by various parties applying the standard and that a particular party may assume several different roles.

5.2.5.5 The supplier shall interface with the independent verification, validation, or test agent as specified in the contract and project plans.

GUIDANCE:

The acquirer or supplier may employ independent verification and validation (IV&V) or test agents. These agents may evaluate the work of the suppliers and subcontractor(s). To ensure that IV&V and test agent resources are used most effectively, it is important for these agents to receive the proper training and experience in the processes, tools, and techniques used by the organization evaluated.

5.2.5.6 The supplier shall interface with other parties as specified in the contract and project plans.

5.2.6 Review and evaluation. This activity consists of the following tasks:

5.2.6.1 The supplier should coordinate contract review activities, interfaces, and communication with the acquirer's organization.

GUIDANCE:

The supplier should conduct the activities in 5.2.6.1 as defined in the project management plan(s).

5.2.6.2 The supplier shall conduct or support the informal meetings, acceptance review, acceptance testing, joint reviews, and audits with the acquirer as specified in the contract and project plans. The joint reviews shall be conducted in accordance with 6.6, audits in accordance with 6.7.

5.2.6.3 The supplier shall perform verification and validation in accordance with 6.4 and 6.5 respectively to demonstrate that the software products or services and processes fully satisfy their respective requirements.

5.2.6.4 The supplier shall make available to the acquirer the reports of evaluation, reviews, audits, testing, and problem resolutions as specified in the contract.

5.2.6.5 The supplier shall provide the acquirer access to the supplier's and subcontractors' facilities for review of software products or services as specified in the contract and project plans.

GUIDANCE:

If the project utilizes multifunctional teams [e.g., within the supplier's organization or between the supplier and subcontractor(s)], the acquirer's participation should be clearly identified and understood by all team participants. For example, the acquirer may be a passive observer of all team activities, occasional visitor, active participant, or chairman and final authority of all team decisions.

5.2.6.6 The supplier shall perform quality assurance activities in accordance with 6.3.

5.2.7 Delivery and completion. This activity consists of the following tasks:

5.2.7.1 The supplier shall deliver the software product or service as specified in the contract.

5.2.7.2 The supplier shall provide assistance to the acquirer in support of the delivered software product or service as specified in the contract.

GUIDANCE:

The supplier should record the transition strategy and approach in the project management plan.

5.3 Development process

The Development Process contains the activities and tasks of the developer. The process contains the activities for requirements analysis, design, coding, integration, testing, and installation and acceptance related to software products. It may contain system related activities if stipulated in the contract. The developer performs or supports the activities in this process in accordance with the contract.

The developer manages the Development Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4). When the developer is the supplier of the developed software product, the developer performs the Supply Process (5.2).

List of activities: This process consists of the following activities:

1) Process implementation;
2) System requirements analysis;
3) System architectural design;
4) Software requirements analysis;
5) Software architectural design;
6) Software detailed design;
7) Software coding and testing;
8) Software integration;
9) Software qualification testing;
10) System integration;
11) System qualification testing;
12) Software installation;
13) Software acceptance support.

GUIDANCE:

1—Preferably, a project's implementation of IEEE/EIA 12207.0 should be specified with respect to the organization's claim of compliance to IEEE/EIA 12207.0. See annex B of this guide for further guidance on compliance.

2—If a system or software item is developed in multiple builds, activities within a process may not be complete until the final build. The developer's plans should identify (1) overall plans for each activity for the contract, (2) detailed planning for each activity for the current build, and (3) planning for future builds covered under the contract to a level of detail compatible with the information available.

3—IEEE/EIA 12207.0 uses "requirement" in many contextual ways. When a qualifier is provided (e.g., acquisition, system, software, test, contract), the qualifier implies the source(s) and receiver(s) of that requirement. When requirement is used without a qualifier, the meaning of that requirement is local to the specific activity or task.

4—In the system activities, (5.3.2 - System requirements analysis, 5.3.3 - System architectural design, 5.3.10 - System integration, and 5.3.11 - System qualification testing), the developer is tasked to perform or support as required by the contract. As discussed in the Acquisition Process (5.1), the acquirer may elect to perform system requirements analysis activities. In similar fashion, the acquirer may (a) elect to perform system design, integration, and qualification testing; (b) task the supplier to perform these activities; or (c) employ other organizations to perform them.

5—Regardless of who actually performs or supports these activities, the role of the software professionals typically varies depending on the type of system developed. If the software is part of a hardware-software system, the software professionals take part in the system activities as described in plans. If the software (possibly with its computer hardware) is considered to constitute a system, or only the software in an existing hardware-software system is changed, the software professional is responsible for performing or supporting these system activities.

6—See IEEE/EIA 12207.1 for guidance on documentation of life cycle data.

7—Figure 4 represents the development process. The following abbreviations are used in figure 4: ACCEPT = acceptance; ARCH = architectural; HW = hardware; MO = manual operations; JT = joint; QUALIF = qualification; REQ = requirements; SW = software; SYS = system.

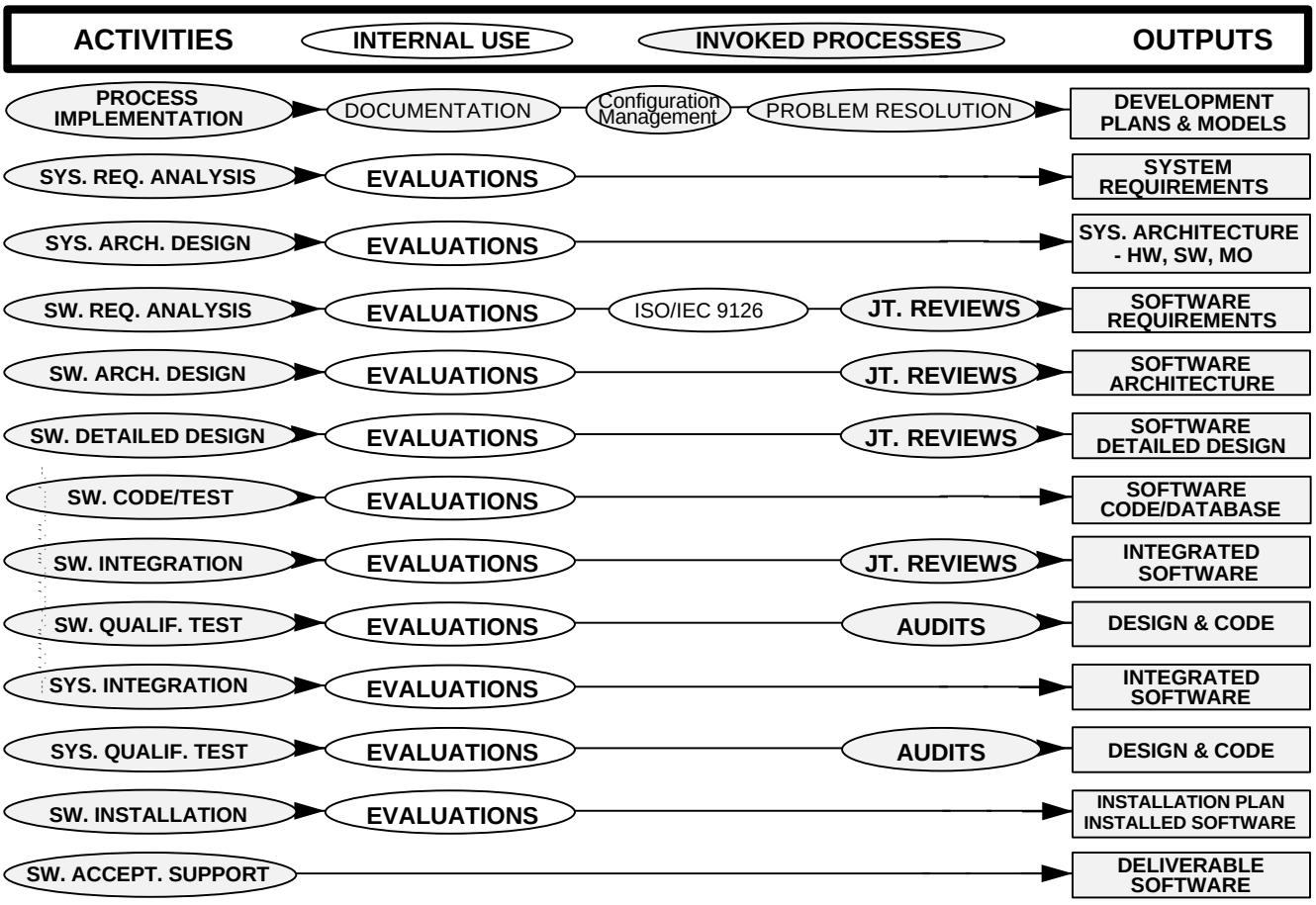

Figure 4—Development process

5.3.1 Process implementation. This activity consists of the following tasks:

5.3.1.1 If not stipulated in the contract, the developer shall define or select a software life cycle model appropriate to the scope, magnitude, and complexity of the project. The activities and tasks of the Development Process shall be selected and mapped onto the life cycle model.

NOTE—These activities and tasks may overlap or interact and may be performed iteratively or recursively.

GUIDANCE:

If not proposed by the acquirer or established by the supplier, the developer should establish an appropriate life cycle model(s) before continuing with the development process. The details of the mapping of the development process onto the life cycle model(s) should include activities and tasks to be conducted, when the activities and tasks would be conducted, and person(s) responsible for conducting the activities and tasks. Additional guidance on use of software life cycle models may be found in IEEE Std 1074.

5.3.1.2 The developer shall:

a) Document the outputs in accordance with the Documentation Process (6.1).

b) Place the outputs under the Configuration Management Process (6.2) and perform change control in accordance with it.

c) Document and resolve problems and nonconformances found in the software products and tasks in accordance with the Problem Resolution Process (6.8).

d) Perform the supporting processes (clause 6) as specified in the contract.

e) Establish baselines for each configuration item at appropriate times, as determined by the acquirer and the supplier.

GUIDANCE:

1—For 5.3.1.2 a), implementation of the documentation process is the recording of information in any media and may include the preparation of paper documents. Rationale for key decisions made in conducting the development of the system/software should be documented. The rationale should include trade-offs considered, analysis methods, and criteria used to make those decisions. The meaning of “key decisions” and the approach for providing the rationale should be described in the plans. The development and recording of information is an intrinsic part of the development process. The documentation process addresses the presentation and management of this information. The information generated during the development process should comply with the objectives of annex H of IEEE/EIA 12207.0. Guidance on the format and content of information developed is provided in IEEE/EIA 12207.1.

2—For 5.3.1.2 b), the developer should consider including different states through which identified items, products, or entities pass as part of configuration control (e.g., author control, developer’s project control, acquirer control). These states can then determine when other processes may be employed. (For example, only products at project-level-or-higher configuration control may be listed in configuration status accounting records and employ the problem resolution process to officially report and track any problems.) The developer should also document the persons or groups with the authority to authorize changes and to make changes at each level, if applicable. The developer should document the steps to be followed to request authorization for changes, process change requests, track changes, distribute changes, and preserve past versions. Changes that affect an item, product, or entity already under acquirer control should be proposed to the acquirer in accordance with contractually established forms and procedures, if any. Baselines may be used during the development process as a form of control of a configuration item. (Additional information on baselines may be found in E.14 of IEEE/EIA 12207.0.)

5.3.1.3 The developer shall select, tailor, and use those standards, methods, tools, and computer programming languages (if not stipulated in the contract) that are documented, appropriate, and established by the organization for performing the activities of the Development Process and supporting processes (clause 6).

GUIDANCE:

A developer’s selection of standards, methods, tools, and computer languages may include selection of the developer’s own standard practices, including software reuse practices. (See annex F of this guide for further information on reusable software products.)

5.3.1.4 The developer shall develop plans for conducting the activities of the development process. The plans should include specific standards, methods, tools, actions, and responsibility associated with the development and qualification of all requirements including safety and security. If necessary, separate plans may be developed. These plans shall be documented and executed.

GUIDANCE:

1—The developer may reference, rather than include, specific standards, methods, tools, practices, and computer programming languages in the plans for conducting the activities.

2—Planning for development describes the approach (methods/procedures/tools) to applicable activities and tasks of the development process, covers all applicable clauses regarding development, identifies applicable risks/uncertainties regarding those activities and tasks, and describes plans for dealing with the risks/uncertainties.

3—Software development planning should be based on the software life cycle model (see 5.3.1.1).

4—Plans to be included may be project management plans and/or software development plans.

5.3.1.5 Non-deliverable items may be employed in the development or maintenance of the software product. However, it shall be ensured that the operation and maintenance of the deliverable software product after its delivery to the acquirer are independent of such items, otherwise those items should be considered as deliverable.

GUIDANCE:

The developer should bring to the attention of the supplier (and to the acquirer) those nondeliverable items that would be needed during operations and maintenance of the deliverable products to allow the supplier and the acquirer to negotiate usage or delivery of the deliverable products.

5.3.2 System requirements analysis. This activity consists of the following tasks, which the developer shall perform or support as required by the contract:

5.3.2.1 The specific intended use of the system to be developed shall be analyzed to specify system requirements. The system requirements specification shall describe: functions and capabilities of the system; business, organizational and user requirements; safety, security, human-factors engineering (ergonomics), interface, operations, and maintenance requirements; design constraints and qualification requirements. The system requirements specification shall be documented.

GUIDANCE:

1—To fully understand what is required of the intended product, the system requirements analysis activity should include elicitation from the user community.

2—If a system consists of subsystems, the activities in 5.3.2 - System requirements analysis, should be performed iteratively with the activities in 5.3.3 - System architectural design, to define system requirements, design the system and identify subsystems, define the requirements for those subsystems, design the subsystems and identify their components, and so on.

3—Each requirement should be stated in such a way that an objective test can be defined for it.

4—The developer should analyze acquisition requirements concerning computer hardware resource utilization (e.g., maximum allowable use of processor capacity, memory capacity, input/output device capacity, auxiliary storage device capacity, communications/network equipment capacity). If there are no acquisition requirements concerning computer hardware resource utilization, or they are very general, the developer should establish appropriate utilization requirements as part of the system requirements activity. Establishing utilization requirements should be an activity iterated with design activities.

5—The definition of system safety and security requirements should also include what the system must not do.

5.3.2.2 The system requirements shall be evaluated considering the criteria listed below. The results of evaluations shall be documented.

a) Traceability to acquisition needs;
b) Consistency with acquisition needs;
c) Testability;
d) Feasibility of system architectural design;
e) Feasibility of operation and maintenance.

GUIDANCE:

The requirements to perform evaluations 5.3.2.2 d) and 5.3.2.2 e) are intended to task the developer to determine the feasibility of creating a system architectural design based on the requirements and the feasibility of operating and maintaining such a system. It may be necessary to perform selected aspects of design, such as prototyping or simulation, to determine whether it is feasible to develop a product(s) meeting those requirements.

5.3.3 System architectural design. This activity consists of the following tasks, which the developer shall perform or support as required by the contract:

5.3.3.1 A top-level architecture of the system shall be established. The architecture shall identify items of hardware, software, and manual-operations. It shall be ensured that all the system requirements are allocated among the items. Hardware configuration items, software configuration items, and manual operations shall be subsequently identified from these items. The system architecture and the system requirements allocated to the items shall be documented.

GUIDANCE:

1—IEEE/EIA 12207.0 places few requirements upon the goals or conduct of system architectural design, recognizing that this area of technology and practice is undergoing rapid evolution. The following guidance may be helpful in selecting a method and a set of goals.

2—In many cases, it is appropriate to view the system architecture as a mechanism for facilitating communications between the acquirer, the developer, the supplier, and other key stakeholders regarding clarification of requirements and their impact on the structure of the system. Although such a usage is not a requirement of the standard, it does contribute to the achievement of the evaluation criteria listed in 5.3.3.2.

3—IEEE/EIA 12207.0 does not require the use of any particular architectural method, e.g., functional decomposition. That selection should be made based upon the characteristics of the development effort. In some cases, it is appropriate to view the software architectural efforts as an extension of the system architectural effort, continuing at a greater level of detail and with greater attention to software-specific concerns.

4—IEEE/EIA 12207.0 does not require any particular output from the architectural effort aside from identification of items and allocation of requirements. It is often appropriate (and supportive of the evaluation criteria) to use the architecture as a means for formulating and recording system-wide design decisions (that is, decisions about the system's concept of execution and interface characteristics, and other decisions affecting the selection and design of system items). The results of such decisions may be recorded in the form of the system-wide design. Results of decisions regarding item identification and requirements traceability may be recorded in the form of the architectural design and traceability cross reference.

5—In many cases, architectural design should be regarded as an iterative activity. Final identification of all items and allocation of all requirements may be completed during iterations other than the first one.

6—The top-level architectural design of a system (i.e., the first partition) may include subsystems. The system architectural design may incorporate alternative views of architecture.

5.3.3.2 The system architecture and the requirements for the items shall be evaluated considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the system requirements;
b) Consistency with the system requirements;
c) Appropriateness of design standards and methods used;
d) Feasibility of the software items fulfilling their allocated requirements;
e) Feasibility of operation and maintenance.

GUIDANCE:

The requirements to perform evaluations 5.3.3.2 d) and 5.3.3.2 e) are intended to task the developer to determine the feasibility of creating a system architectural design based on the requirements and the feasibility of operating and maintaining such a system. It may be necessary to perform selected aspects of software requirements analysis to determine whether it is feasible to develop a product(s) meeting those requirements. For 5.3.3.2 d), the developer should consider whether the planned and potential computer resources (processors, etc.) would be adequate to satisfy the requirements.

5.3.4 Software requirements analysis. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.4.1 The developer shall establish and document software requirements, including the quality characteristics specifications, described below. Guidance for specifying quality characteristics may be found in ISO/IEC 9126.

a) Functional and capability specifications, including performance, physical characteristics, and environmental conditions under which the software item is to perform;

b) Interfaces external to the software item;

c) Qualification requirements;

d) Safety specifications, including those related to methods of operation and maintenance, environmental influences, and personnel injury;

e) Security specifications, including those related to compromise of sensitive information;

f) Human-factors engineering (ergonomics), including those related to manual operations, human-equipment interactions, constraints on personnel, and areas needing concentrated human attention, that are sensitive to human errors and training;

g) Data definition and database requirements;

h) Installation and acceptance requirements of the delivered software product at the operation and maintenance site(s);

i) User documentation;

j) User operation and execution requirements;

k) User maintenance requirements.

GUIDANCE:

1—ISO/IEC 9126 identifies six quality characteristics to be included: functionality, reliability, usability, efficiency, maintainability, and portability.

2—All items, 5.3.4.1 a) through 5.3.4.1 k), should be stated in such a way that objective criteria may be defined for them.

3—Guidance regarding software safety plans may be found in IEEE Std 1228.

5.3.4.2 The developer shall evaluate the software requirements considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to system requirements and system design;
b) External consistency with system requirements;
c) Internal consistency;
d) Testability;
e) Feasibility of software design;
f) Feasibility of operation and maintenance.

GUIDANCE:

The requirements to perform evaluations 5.3.4.2 e) and 5.3.4.2 f) are intended to task the developer to determine the feasibility of creating a software architectural design for the software item based on the requirements and the feasibility of operating and maintaining a system containing such a software item. It may be necessary to perform selected aspects of design, such as prototyping or simulation, to determine whether it is feasible to develop a product(s) meeting those requirements.

5.3.4.3 The developer shall conduct joint review(s) in accordance with 6.6.

GUIDANCE:

Conducting joint review(s) includes planning and taking part in joint technical and management reviews. Annex G of this guide describes a candidate set of joint management reviews that may be held during a software development project. IEEE Std 1028 may be helpful in implementing this requirement.

5.3.5 Software architectural design. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.5.1 The developer shall transform the requirements for the software item into an architecture that describes its top-level structure and identifies the software components. It shall be ensured that all the requirements for the software item are allocated to its software components and further refined to facilitate detailed design. The architecture of the software item shall be documented.

GUIDANCE:

1—In many cases, it is appropriate to view the software architecture as a mechanism for facilitating communications among the system designers, the software designers, and other key stakeholders regarding clarification of requirements and their impact on the structure of the software.

2—The architectural activity is concerned with creating a strong overall structure for the software. During this activity, attention should be paid to the structural mechanisms of the system architecture. For example, architectural design should produce common mechanisms for managing persistent objects, providing a user interface, and managing storage.

3—It is often appropriate (and supportive of the evaluation criteria) to use the architecture as a means for formulating and recording software design decisions (that is, decisions about the software's behavioral design and other decisions affecting the selection and design of software components). The results of such decisions may be recorded in the form of the software item design. Results of decisions regarding software component identification and requirements traceability may be recorded in the form of the architectural design and a traceability cross reference.

4—The developer's description of the software architecture includes the description of the concept of execution among the software components.

5—There is no universal agreement on how much design detail should be documented and reviewed as the developer's "top-level structure" for each software item. Therefore, the developer's standards, methods, and tools, described in the development plans, should clearly articulate the approach to creating software architectural and detailed designs. In particular, the plans should address (1) standards, methods, and tools used; (2) terminology and graphic notations to describe different design elements; (3) level of detail associated with architectural and detailed design; and (4) approach to reviewing design information with project

participants. To avoid misconceptions by various project stakeholders, it's important that these design-related issues be resolved, documented, and clearly understood.

6—The term "software component" in IEEE/EIA 12207.0 represents an entity that is refined into lower levels containing software units and is tested as part of software integration. The development plan(s) should clearly state the approach to creating software top-level structure, architectural and detailed design, software coding, and all levels of software testing. In particular, the plans should address (1) standards, methods, and tools used; (2) terminology and graphic notations to describe different design elements; (3) level of detail associated with architectural and detailed design; (4) relationship of software architectural design to detailed design; (5) approach to reviewing design information with project participants; and (6) approach to software integration testing. To avoid misconceptions by various project stakeholders, it's important that these design/integration/test issues be resolved, documented, and clearly understood.

7—The software architecture should be consistent with the system architecture.

5.3.5.2 The developer shall develop and document a top-level design for the interfaces external to the software item and between the software components of the software item.

GUIDANCE:

The developer's interface design description should include the description of the interface characteristics of the software items, external entities, and software components.

5.3.5.3 The developer shall develop and document a top-level design for the database.

5.3.5.4 The developer should develop and document preliminary versions of user documentation.

GUIDANCE:

Since many systems involve interaction with people, the developer frequently plans and prepares user documentation. The developer should communicate with the user regarding all issues that impact the user (e.g., requirements, design, test, user documentation issues). Preliminary versions of user documentation may be prepared as part of the software architectural design activity and updated during successive activities. It is important to ensure that all affected parties understand (1) how user input into the development process should be collected and consolidated, (2) what user documentation should be prepared, (3) when preliminary and updated information should be due, and (4) what parties should be responsible for preparing/reviewing/updating the information. Approval of user documentation should be covered in the acquirer-supplier agreement.

5.3.5.5 The developer shall define and document preliminary test requirements and the schedule for Software Integration.

GUIDANCE:

The developer should determine the appropriate time in the development process to begin planning for software integration and testing. The developer may define and document preliminary test requirements and the schedule for software integration as part of the software architectural design activity; or document the test information during another part of the development process. It is important to ensure that all affected parties understand (1) what test documentation should be prepared, (2) when preliminary and updated information should be due, and (3) what parties should be responsible for preparing/reviewing/updating the information.

5.3.5.6 The developer shall evaluate the architecture of the software item and the interface and database designs considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the requirements of the software item;
b) External consistency with the requirements of the software item;
c) Internal consistency between the software components;
d) Appropriateness of design methods and standards used;
e) Feasibility of detailed design;

f) Feasibility of operation and maintenance.

GUIDANCE:

When evaluating and documenting the results of these evaluations, the developer should account for the feasibility of the architecture to implement the requirements and satisfy user needs.

5.3.5.7 The developer shall conduct joint review(s) in accordance with 6.6.

GUIDANCE:

Conducting joint review(s) includes planning and taking part in joint technical and management reviews. Annex G of this guide describes a candidate set of joint management reviews that may be held during a software development project. IEEE Std 1028 may be helpful in implementing this requirement.

5.3.6 Software detailed design. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.6.1 The developer shall develop a detailed design for each software component of the software item. The software components shall be refined into lower levels containing software units that can be coded, compiled, and tested. It shall be ensured that all the software requirements are allocated from the software components to software units. The detailed design shall be documented.

GUIDANCE:

1—The developer's detailed design description should include a description of each software unit of the software item.

2—The supplier's standards, methods, and tools, described in the development plan(s), should clearly articulate the approach to creating detailed design, software coding, and all levels of software testing. In particular, the plan(s) should address (1) standards, methods, and tools used; (2) terminology and graphic notations to describe different design elements; (3) level of detail associated with detailed design; (4) relationship of detailed design to separately compilable software entities and to separately executable software entities; (5) approach to reviewing design information with project participants; and (6) approach to unit testing. To avoid misconceptions by various project stakeholders, it's important that these design/code/test issues be resolved, documented, and clearly understood.

5.3.6.2 The developer shall develop and document a detailed design for the interfaces external to the software item, between the software components, and between the software units. The detailed design of the interfaces shall permit coding without the need for further information.

5.3.6.3 The developer shall develop and document a detailed design for the database.

GUIDANCE:

The developer should describe each software unit used for database access or manipulation and should describe data elements and data element assemblies of the database.

5.3.6.4 The developer shall update user documentation as necessary.

5.3.6.5 The developer shall define and document test requirements and schedule for testing software units. The test requirements should include stressing the software unit at the limits of its requirements.

GUIDANCE:

The developer should determine the appropriate time in the development process to begin planning for software unit testing. The developer may define and document test requirements and the schedule for software unit testing as part of the software detailed design activity or create and document the test information during another part of the development process. It is important to ensure that all affected parties understand (1) what test documentation should be prepared, (2) when information should be due, and (3) what parties should be responsible for preparing/reviewing/updating the information.

5.3.6.6 The developer shall update the test requirements and the schedule for Software Integration.

GUIDANCE:

The developer should determine the appropriate time in the development process to begin or update planning for software integration and testing. The developer may update test requirements and the schedule for software integration as part of the software detailed design activity or update the test information during another part of the development process. It's important that all project stakeholders clearly understand (1) what test documentation should be prepared, (2) when preliminary and updated information should be due, and (3) what parties should be responsible for preparing/reviewing/updating the information.

5.3.6.7 The developer shall evaluate the software detailed design and test requirements considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the requirements of the software item;
b) External consistency with architectural design;
c) Internal consistency between software components and software units;
d) Appropriateness of design methods and standards used;
e) Feasibility of testing;
f) Feasibility of operation and maintenance.

5.3.6.8 The developer shall conduct joint review(s) in accordance with 6.6.

GUIDANCE:

Conducting joint review(s) includes planning and taking part in joint technical and management reviews. Annex G of this guide describes a candidate set of joint management reviews that may be held during a software development project. IEEE Std 1028 may be helpful in implementing this requirement.

5.3.7 Software coding and testing. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.7.1 The developer shall develop and document the following:

a) Each software unit and database;
b) Test procedures and data for testing each software unit and database;

GUIDANCE:

1—Software units may be defined in many ways. The definition(s) to be used should be determined by the developing organization. (See guidance in clause 3 of this guide for more information on possible definitions of software unit.)

2—Database development may include database activities, such as preparing data and populating the database or other data files with data values.

5.3.7.2 The developer shall test each software unit and database ensuring that it satisfies its requirements. The test results shall be documented.

5.3.7.3 The developer shall update the user documentation as necessary.

5.3.7.4 The developer shall update the test requirements and the schedule for Software Integration.

GUIDANCE:

The developer should determine the appropriate time in the development process to begin or update planning for software integration and testing. The developer may update test requirements and the schedule for software integration as part of the software coding and testing activity or update the test information during another part of the development process. It is important to ensure that all affected parties understand (1) what test documentation should be prepared, (2) when preliminary and updated information should be due, and (3) what parties should be responsible for preparing/reviewing/updating the information.

5.3.7.5 The developer shall evaluate software code and test results considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the requirements and design of the software item;
b) External consistency with the requirements and design of the software item;
c) Internal consistency between unit requirements;
d) Test coverage of units;
e) Appropriateness of coding methods and standards used;
f) Feasibility of software integration and testing;
g) Feasibility of operation and maintenance.

GUIDANCE:

The requirement to evaluate the appropriateness of coding methods and standards is intended to task the developer to evaluate the appropriate use of coding methods and standards.

5.3.8 Software integration. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.8.1 The developer shall develop an integration plan to integrate the software units and software components into the software item. The plan shall include test requirements, procedures, data, responsibilities, and schedule. The plan shall be documented.

GUIDANCE:

1—The plan should include, as applicable, a description of the test environment, rationale, and data recording and analysis procedures. The plan should also include any integration and test information that was developed or updated during other development activities. (See 5.3.5.5, 5.3.6.6, 5.3.7.4.)

2—See IEEE/EIA 12207.1 for guidance on software integration plan.

5.3.8.2 The developer shall integrate the software units and software components and test as the aggregates are developed in accordance with the integration plan. It shall be ensured that each aggregate satisfies the requirements of the software item and that the software item is integrated at the conclusion of the integration activity. The integration and test results shall be documented.

5.3.8.3 The developer shall update the user documentation as necessary.

5.3.8.4 The developer shall develop and document, for each qualification requirement of the software item, a set of tests, test cases (inputs, outputs, test criteria), and test procedures for conducting Software Qualification Testing. The developer shall ensure that the integrated software item is ready for Software Qualification Testing.

GUIDANCE:

1—The developer should identify the software or system requirements addressed by each test case and should ensure that all software requirements are included as part of software qualification testing.

2—In IEEE/EIA 12207.0, a test consists of one or more test cases.

3—The developer should determine the appropriate time in the development process to begin or update planning for software qualification testing. The developer may develop and document qualification test details as part of the software integration activity or during another part of the development process. It is important to ensure that all affected parties understand (1) what test documentation should be prepared, (2) when preliminary and updated information should be due, and (3) what parties should be responsible for preparing/reviewing/updating the information.

4—The developer should include pretest procedures for hardware and software required as part of the test preparations.

5—If the acquirer plans to witness qualification testing, the developer should conduct a dry run of all qualification test cases and procedures to ensure the qualification test documentation is correct and the software performs as expected.

5.3.8.5 The developer shall evaluate the integration plan, design, code, tests, test results, and user documentation considering the criteria listed below. The results of the evaluations shall be documented.

a) Traceability to the system requirements;
b) External consistency with the system requirements;
c) Internal consistency;
d) Test coverage of the requirements of the software item;
e) Appropriateness of test standards and methods used;
f) Conformance to expected results;
g) Feasibility of software qualification testing;
h) Feasibility of operation and maintenance.

5.3.8.6 The developer shall conduct joint review(s) in accordance with 6.6.

GUIDANCE:

Conducting joint review(s) includes planning and taking part in joint technical and management reviews. Annex G of this guide describes a candidate set of joint management reviews that may be held during a software development project. IEEE Std 1028 may be helpful in implementing this requirement.

5.3.9 Software qualification testing. For each software item (or software configuration item, if identified), this activity consists of the following tasks:

5.3.9.1 The developer shall conduct qualification testing in accordance with the qualification requirements for the software item. It shall be ensured that the implementation of each software requirement is tested for compliance. The qualification testing results shall be documented.

GUIDANCE:

The developer should conduct qualification testing in accordance with the documented test cases (inputs, outputs, pass/fail criteria) and test procedures. The developer should document the test results, including the completion status of each test case, and an overall assessment of the software, identifying remaining deficiencies/limitations/constraints and associated impacts as well as recommendations for correction. The developer should also indicate how the test environment may be different from the operational environment and the respective effect on test results.

5.3.9.2 The developer shall update the user documentation as necessary.

5.3.9.3 The developer shall evaluate the design, code, tests, test results, and user documentation considering the criteria listed below. The results of the evaluations shall be documented.

a) Test coverage of the requirements of the software item;
b) Conformance to expected results;
c) Feasibility of system integration and testing, if conducted;
d) Feasibility of operation and maintenance.

5.3.9.4 The developer shall support audit(s) in accordance with 6.7. The results of the audits shall be documented. If both hardware and software are under development or integration, the audits may be postponed until the System Qualification Testing.

5.3.9.5 Upon successful completion of the audits, if conducted, the developer shall:

a) Update and prepare the deliverable software product for System Integration, System Qualification Testing, Software Installation, or Software Acceptance Support as applicable.

NOTE—The Software Qualification Testing may be used in the Verification Process (6.4) or the Validation Process (6.5).

5.3.10 System integration. This activity consists of the following tasks, which the developer shall perform or support as required by the contract.

5.3.10.1 The software configuration items shall be integrated, with hardware configuration items, manual operations, and other systems as necessary, into the system. The aggregates shall be tested, as they are developed, against their requirements. The integration and the test results shall be documented.

5.3.10.2 For each qualification requirement of the system, a set of tests, test cases (inputs, outputs, test criteria), and test procedures for conducting System Qualification Testing shall be developed and documented. The developer shall ensure that the integrated system is ready for System Qualification Testing.

GUIDANCE:

1—The developer should determine the appropriate time in the development process to begin or update planning for system qualification testing. The developer may develop and document qualification test details as part of the system integration activity or during another part of the development process. It is important to ensure that all affected parties understand (1) what test documentation should be prepared, (2) when preliminary and updated information should be due, and (3) what parties should be responsible for preparing/reviewing/updating the information.

2—The developer should include pretest procedures for hardware and software required as part of the test preparations.

3—If the acquirer plans to witness qualification testing, the developer should conduct a dry run of all qualification test cases and procedures to ensure the qualification test documentation is correct and the system performs as expected.

5.3.10.3 The integrated system shall be evaluated considering the criteria listed below. The results of the evaluations shall be documented.

a) Test coverage of system requirements;
b) Appropriateness of test methods and standards used;
c) Conformance to expected results;
d) Feasibility of system qualification testing;
e) Feasibility of operation and maintenance.

5.3.11 System qualification testing. This activity consists of the following tasks, which the developer shall perform or support as required by the contract.

5.3.11.1 System qualification testing shall be conducted in accordance with the qualification requirements specified for the system. It shall be ensured that the implementation of each system requirement is tested for compliance and that the system is ready for delivery. The qualification testing results shall be documented.

5.3.11.2 The system shall be evaluated considering the criteria listed below. The results of the evaluations shall be documented.

a) Test coverage of system requirements;
b) Conformance to expected results;
c) Feasibility of operation and maintenance.

5.3.11.3 The developer shall support audit(s) in accordance with 6.7. The results of the audit(s) shall be documented.

NOTE—This subclause is not applicable to those software configuration items for which audits were conducted previously.

5.3.11.4 Upon successful completion of the audit(s), if conducted, the developer shall:

a) Update and prepare the deliverable software product for Software Installation and Software Acceptance Support.

NOTE—The System Qualification Testing may be used in the Verification Process (6.4) or the Validation Process (6.5).

GUIDANCE:

The developer should define and record information needed to operate and maintain the software.

5.3.12 Software installation. This activity consists of the following tasks:

5.3.12.1 The developer shall develop a plan to install the software product in the target environment as

designated in the contract. The resources and information necessary to install the software product shall be determined and be available. As specified in the contract, the developer shall assist the acquirer with the set-up activities. Where the installed software product is replacing an existing system, the developer shall support any parallel running activities that are required by contract. The installation plan shall be documented.

GUIDANCE:

The target environment may be the user site, the support site, or the operational site.

5.3.12.2 The developer shall install the software product in accordance with the installation plan. It shall be ensured that the software code and databases initialize, execute, and terminate as specified in the contract. The installation events and results shall be documented.

5.3.13 Software acceptance support. This activity consists of the following tasks:

5.3.13.1 The developer shall support the acquirer's acceptance review and testing of the software product. Acceptance review and testing shall consider the results of the Joint Reviews (6.6), Audits (6.7), Software Qualification Testing, and System Qualification Testing (if performed). The results of the acceptance review and testing shall be documented.

GUIDANCE:

Further information on acquirer acceptance may be found in the guidance to 5.1.5.2.

5.3.13.2 The developer shall complete and deliver the software product as specified in the contract.

GUIDANCE:

The delivery of the software product should include, as applicable, delivery of the source files and executable software, including batch files, command files, or other files needed to regenerate the executable software; a description of packaging requirements; compilation/build procedures for creating executable software; procedures for modifying the software; measured hardware resource utilization of the "as built" software; and a description of the software version.

5.3.13.3 The developer shall provide initial and continuing training and support to the acquirer as specified in the contract.

No further guidance provided.

5.4 Operation process

The Operation Process contains the activities and tasks of the operator. The process covers the operation of the software product and operational support to users. Because operation of software product is integrated into the operation of the system, the activities and tasks of this process refer to the system.

The operator manages the Operation Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4). When the operator is the supplier of the operation service, the operator performs the Supply Process (5.2).

List of activities. This process consists of the following activities:

1) Process implementation;
2) Operational testing;
3) System operation;
4) User support.

GUIDANCE:

1—Preferably, a project's implementation of IEEE/EIA 12207.0 should be specified with respect to the organization's claim of compliance to IEEE/EIA 12207.0. See annex B of this guide for further guidance on compliance.

2—IEEE/EIA 12207.0 uses "requirement" in many contextual ways. When a qualifier is provided (e.g., acquisition, system, software, test, contract), the qualifier implies the source(s) and receiver(s) of that requirement. When requirement is used without a qualifier, the meaning of that requirement is local to the specific activity or task.

3—The operation process is only applicable to systems that utilize "operators" running software products for end users. End users may be instructors and students "using" a software product at a simulation center; those "using" client server network programs run by system administrators; and/or those "using" the reports or other output produced by an operator. In many systems, the operation process may not be applicable; either the end users interact directly ("hands on") with the software products (e.g., pilots in a jet plane) or the software product runs with minimal human intervention (e.g., a process control system in a manufacturing facility).

4—Figure 5 represents the operation process.

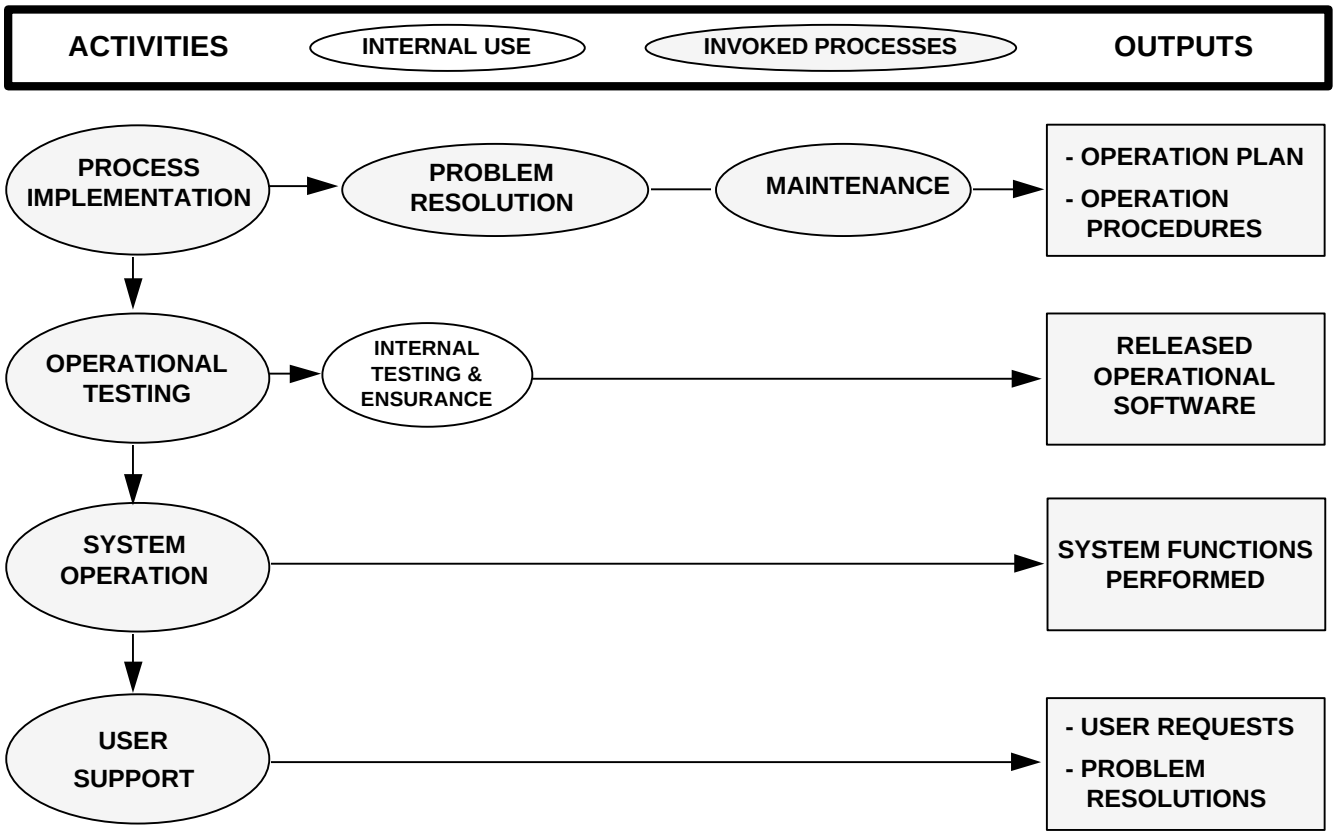

Figure 5—Operation process

5.4.1 Process implementation. This activity consists of the following tasks:

5.4.1.1 The operator shall develop a plan and set operational standards for performing the activities and tasks of this process. The plan shall be documented and executed.

5.4.1.2 The operator shall establish procedures for receiving, recording, resolving, tracking problems, and providing feedback. Whenever problems are encountered, they shall be recorded and entered into the Problem Resolution Process (6.8).

5.4.1.3 The operator shall establish procedures for testing the software product in its operation environment, for entering problem reports and modification requests to the Maintenance Process (5.5), and for releasing the software product for operational use.

GUIDANCE:

The test procedures used may be procedures previously developed by the developer, acquirer, or other party.

5.4.2 Operational testing. This activity consists of the following tasks:

5.4.2.1 For each release of the software product, the operator shall perform operational testing, and, on satisfying the specified criteria, release the software product for operational use.

5.4.2.2 The operator shall ensure that the software code and databases initialize, execute, and terminate as described in the plan.

5.4.3 System operation. This activity consists of the following task:

5.4.3.1 The system shall be operated in its intended environment according to the user documentation.

5.4.4 User support. This activity consists of the following tasks:

5.4.4.1 The operator shall provide assistance and consultation to the users as requested. These requests and subsequent actions shall be recorded and monitored.

5.4.4.2 The operator shall forward user requests, as necessary, to the Maintenance Process (clause 5.5) for resolution. These requests shall be addressed and the actions that are planned and taken shall be reported to the originators of the requests. All resolutions shall be monitored to conclusion.

5.4.4.3 If a reported problem has a temporary work-around before a permanent solution can be released, the originator of the problem report shall be given the option to use it. Permanent corrections, releases that include previously omitted functions or features, and system improvements shall be applied to the operational software product using the Maintenance Process (5.5).

GUIDANCE:

The operator should coordinate with the maintainer to identify how temporary fixes will be handled and what documentation should accompany any temporary fixes sent to the operator (e.g., problem reports, listings of updated code).

5.5 Maintenance process

The Maintenance Process contains the activities and tasks of the maintainer. This process is activated when the software product undergoes modifications to code and associated documentation due to a problem or the need for improvement or adaptation. The objective is to modify existing software product while preserving its integrity. This process includes the migration and retirement of the software product. The process ends with the retirement of the software product.

The activities provided in this clause are specific to the Maintenance Process; however, the process may utilize other processes in this International Standard. If the Development Process (5.3) is utilized, the term developer there is interpreted as maintainer.

The maintainer manages the Maintenance Process at the project level following the Management Process (7.1), which is instantiated in this process; establishes an infrastructure under the process following the Infrastructure Process (7.2); tailors the process for the project following the Tailoring Process (annex A); and manages the process at the organizational level following the Improvement Process (7.3) and the Training Process (7.4). When the maintainer is the supplier of the maintenance service, the maintainer performs the Supply Process (5.2).

List of activities. This process consists of the following activities:

1) Process implementation;
2) Problem and modification analysis;
3) Modification implementation;
4) Maintenance review/acceptance;
5) Migration;
6) Software retirement.

GUIDANCE:

1—Preferably, a project's implementation of IEEE/EIA 12207.0 should be specified with respect to the organization's claim of compliance to IEEE/EIA 12207.0. See annex B of this guide for further guidance on compliance.

2—IEEE/EIA 12207.0 uses "requirement" in many contextual ways. When a qualifier is provided (e.g., acquisition, system, software, test, contract), the qualifier implies the source(s) and receiver(s) of that requirement. When requirement is used without a qualifier, the meaning of that requirement is local to the specific activity or task.

3—The maintainer should coordinate with the operator to identify how temporary fixes will be handled and what documentation should accompany any temporary fixes sent to the maintainer (e.g., problem reports, listings of updated code).

4—Figure 6 represents the maintenance process.

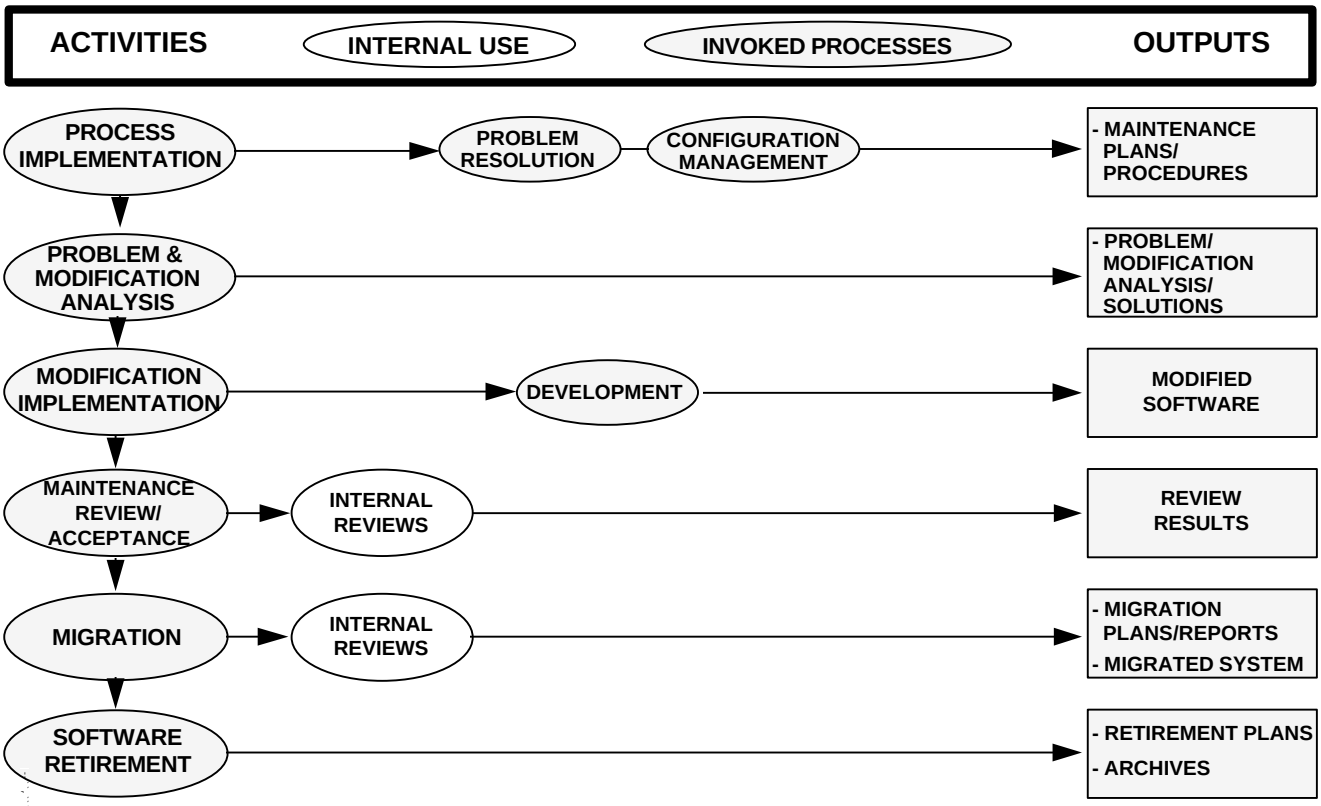

Figure 6—Maintenance process

5.5.1 Process implementation. This activity consists of the following tasks:

5.5.1.1 The maintainer shall develop, document, and execute plans and procedures for conducting the activities and tasks of the Maintenance Process.

5.5.1.2 The maintainer shall establish procedures for receiving, recording, and tracking problem reports and modification requests from the users and providing feedback to the users. Whenever problems are encountered, they shall be recorded and entered into the Problem Resolution Process (6.8).

5.5.1.3 The maintainer shall implement (or establish organizational interface with) the Configuration Management Process (6.2) for managing modifications to the existing system.

5.5.2 Problem and modification analysis. This activity consists of the following tasks:

5.5.2.1 The maintainer shall analyze the problem report or modification request for its impact on the organization, the existing system, and the interfacing systems for the following:

a) Type; for example, corrective, improvement, preventive, or adaptive to new environment;
b) Scope; for example, size of modification, cost involved, time to modify;
c) Criticality; for example, impact on performance, safety, or security.

5.5.2.2 The maintainer shall replicate or verify the problem.

5.5.2.3 Based upon the analysis, the maintainer shall consider options for implementing the modification.

5.5.2.4 The maintainer shall document the problem/modification request, the analysis results, and implementation options.

5.5.2.5 The maintainer shall obtain approval for the selected modification option as specified in the contract.

5.5.3 Modification implementation. This activity consists of the following tasks:

5.5.3.1 The maintainer shall conduct analysis and determine which documentation, software units, and versions thereof need to be modified. These shall be documented.

GUIDANCE:

1—If there is a complete set of up-to-date software-related information (i.e., requirements, design, test, operation, maintenance documentation), the maintainer should determine which documentation should be updated with each maintenance action. However, on many projects, the software-related information is not complete, or the information is obsolete. In such instances, the maintainer should determine (1) how much new information should be created as part of the maintenance action and (2) what existing information should be updated as part of the maintenance action. The amount of information created or updated as part of the maintenance activity depends upon the scope, magnitude, and criticality of the maintenance actions, the available resources in the maintenance environment, the resources available to the acquirer funding the maintenance activity, the impact of the maintenance actions on other project stakeholders, and other factors.

2—The use and definition of the term “software unit” may vary from project to project. It’s important for the maintainer to understand the software design concepts and terminology used on the product undergoing maintenance and to determine which design elements and which versions of software should be modified. The definition(s) of the term “software unit” to be used should be determined by the maintaining organization, taking into consideration the developer’s definition.

5.5.3.2 The maintainer shall enter the Development Process (5.3) to implement the modifications. The requirements of the Development Process shall be supplemented as follows:

a) Test and evaluation criteria for testing and evaluating the modified and the unmodified parts (software units, components, and configuration items) of the system shall be defined and documented.

b) The complete and correct implementation of the new and modified requirements shall be ensured. It also shall be ensured that the original, unmodified requirements were not affected. The test results shall be documented.

GUIDANCE:

1—Not all software and system requirements need to be retested with every software modification. The scope of retesting that should occur to ensure “that the original, unmodified requirements were not affected” may vary based on the magnitude of the change action, the criticality of the affected parts or system functions, the urgency for the update, the release strategy to the user environment, and other factors.

2—Modification of reusable software products (e.g., COTS, previously developed software products) should be carefully considered. Modification may result in the vendor or provider of the product dropping support or requiring a specifically negotiated and priced effort for future support. Additional information on reusable software products may be found in annex F of this guide.

5.5.4 Maintenance review/acceptance. This activity consists of the following tasks:

5.5.4.1 The maintainer shall conduct review(s) with the organization authorizing the modification to determine the integrity of the modified system.

5.5.4.2 The maintainer shall obtain approval for the satisfactory completion of the modification as specified in the contract.

5.5.5 Migration. This activity consists of the following tasks:

5.5.5.1 If a system or software product (including data) is migrated from an old to a new operational environment, it shall be ensured that any software product or data produced or modified during migration are in accordance with this International Standard.

5.5.5.2 A migration plan shall be developed, documented, and executed. The planning activities shall include users. Items included in the plan shall include the following:

a) Requirements analysis and definition of migration;
b) Development of migration tools;
c) Conversion of software product and data;
d) Migration execution;
e) Migration verification;
f) Support for the old environment in the future.

5.5.5.3 Users shall be given notification of the migration plans and activities. Notifications shall include the following:

a) Statement of why the old environment is no longer to be supported;

b) Description of the new environment with its date of availability;

c) Description of other support options available, if any, once support for the old environment has been removed.

GUIDANCE:

For 5.5.5.3 a), support for the old system should be described in the migration plans, including the level of support.

5.5.5.4 Parallel operations of the old and new environments may be conducted for smooth transition to the new environment. During this period, necessary training shall be provided as specified in the contract.

GUIDANCE:

If a separate training environment provides training for users of an operational system, updates to the operational system should trigger updates to the training system.

5.5.5.5 When the scheduled migration arrives, notification shall be sent to all concerned. All associated old environment's documentation, logs, and code should be placed in archives.

5.5.5.6 A post-operation review shall be performed to assess the impact of changing to the new environment. The results of the review shall be sent to the appropriate authorities for information, guidance, and action.

5.5.5.7 Data used by or associated with the old environment shall be accessible in accordance with the contract requirements for data protection and audit applicable to the data.

5.5.6 Software retirement. This activity consists of the following tasks:

NOTE—The software product will be retired on the request of the owner.

5.5.6.1 A retirement plan to remove active support by the operation and maintenance organizations shall be developed and documented. The planning activities shall include users. The plan shall address the items listed below. The plan shall be executed.

a) Cessation of full or partial support after a certain period of time;
b) Archiving of the software product and its associated documentation;
c) Responsibility for any future residual support issues;
d) Transition to new software product, if applicable;
e) Accessibility of archive copies of data.

5.5.6.2 Users shall be given notification of the retirement plans and activities. Notifications shall include the following:

a) Description of the replacement or upgrade with its date of availability;
b) Statement of why the software product is no longer to be supported;
c) Description of other support options available, once support has been removed.

GUIDANCE:

For 5.5.6.2 b), the timetable for phasing out the software product should be included.

5.5.6.3 Parallel operations of the retiring and the new software product should be conducted for smooth transition to the new system. During this period, user training shall be provided as specified in the contract.

5.5.6.4 When the scheduled retirement arrives, notification shall be sent to all concerned. All associated development documentation, logs, and code should be placed in archives, when appropriate.

5.5.6.5 Data used or associated by the retired software product shall be accessible in accordance with the contract requirements for data protection and audit applicable to the data.

No further guidance provided.

6 Supporting processes

This clause defines the following supporting life cycle processes:

1) Documentation process;
2) Configuration management process;
3) Quality assurance process;
4) Verification process;
5) Validation process;
6) Joint review process;
7) Audit process;
8) Problem resolution process.

The activities and tasks in a supporting process are the responsibility of the organization performing that process. This organization ensures that the process is in existence and functional.

The organization employing and performing a supporting process manages it at the project level following the Management Process (7.1); establishes an infrastructure under it following the Infrastructure Process (7.2); tailors it for the project following the Tailoring Process (annex A); and manages it at the organizational level following the Improvement Process (7.3) and the Training Process (7.4). Joint Reviews, Audits, Verification, and Validation may be used as techniques of Quality Assurance.

GUIDANCE:

1—The processes in clause 6 may require the development of plans. These plans may be implemented as separate plans or combined with plans for other processes. As an example, the “Documentation plan” may be part of the “Project plan” in 5.2.4.5.

2—It is not the intent of this guide to suggest that the responsibility for each supporting process should be assigned to a distinct organization. Multiple processes may be assigned to a single organization.

3—The activities within each supporting process may be performed iteratively and in any order deemed appropriate to support the production of the required software product or service. See 1.5 of IEEE/EIA 12207.0, third paragraph, for additional information.

6.1 Documentation process

The Documentation Process is a process for recording information produced by a life cycle process or activity. The process contains the set of activities, which plan, design, develop, produce, edit, distribute, and maintain those documents needed by all concerned such as managers, engineers, and users of the system or software product.

Execution of this process by an organization results in the establishment of internal documentation standards (such as standards for program management plan and software design document) in a suitable media — paper, electronic, or other. The terms used in this process need to be interpreted accordingly for a given media or domain.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Design and development;
3) Production;
4) Maintenance.

GUIDANCE:

1—Although IEEE/EIA 12207.0 only tasks the development process to invoke the documentation process, the intent of IEEE/EIA 12207.0 is to encourage each organization to establish organizational documentation standards to meet its needs.

2—Additional guidance on developing software documentation products may be found in IEEE/EIA 12207.0 H.5, in IEEE/EIA 12207.1, and in EIA/IEEE J-STD-016.

6.1.1 Process implementation. This activity consists of the following task:

6.1.1.1 A plan, identifying the documents to be produced during the life cycle of the software product, shall be developed, documented, and implemented. For each identified document, the following shall be addressed:

a) Title or Name;

b) Purpose;

c) Intended audience;

d) Procedures and responsibilities for inputs, development, review, modification, approval, production, storage, distribution, maintenance, and configuration management;

e) Schedule for intermediate and final versions.

6.1.2 Design and development. This activity consists of the following tasks:

6.1.2.1 Each identified document shall be designed in accordance with applicable documentation standards for format, content description, page numbering, figure/table placement, proprietary/security marking, packaging, and other presentation items.

GUIDANCE:

Documentation should also include or reference conventions needed to understand requirements, design, code, test, or other information. See IEEE/EIA 12207.0 H.5 for further guidance on the presentation form of life cycle data. Additional guidance on documentation of life cycle data may be found in IEEE/EIA 12207.1.

6.1.2.2 The source and appropriateness of input data for the documents shall be confirmed. Automated documentation tools may be used.

6.1.2.3 The prepared documents shall be reviewed and edited for format, technical content, and presentation style against their documentation standards. They shall be approved for adequacy by authorized personnel prior to issue.

6.1.3 Production. This activity consists of the following tasks:

6.1.3.1 The documents shall be produced and provided in accordance with the plan. Production and distribution of documents may use paper, electronic, or other media. Master materials shall be stored in accordance with requirements for record retention, security, maintenance, and backup.

6.1.3.2 Controls shall be established in accordance with the Configuration Management Process (6.2).

6.1.4 Maintenance. This activity consists of the following task:

6.1.4.1 The tasks, that are required to be performed when documentation is to be modified, shall be performed (see 5.5). For those documents that are under configuration management, modifications shall be managed in accordance with the Configuration Management Process (6.2).

No further guidance provided.

6.2 Configuration management process

The Configuration Management Process is a process of applying administrative and technical procedures throughout the software life cycle to: identify and define software items in a system; control modifications and releases of the items; record and report the status of the items and modification requests; ensure the completeness, consistency, and correctness of the items; and control storage, handling, and delivery of the items.

NOTE—When this process is employed on other software products or entities, the term “software item” below is interpreted accordingly.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Configuration identification;
3) Configuration control;
4) Configuration status accounting;
5) Configuration evaluation;
6) Release management and delivery.

6.2.1 Process implementation. This activity consists of the following task:

6.2.1.1 A configuration management plan shall be developed. The plan shall describe: the configuration management activities; procedures and schedule for performing these activities; the organization(s) responsible for performing configuration management and activities; and their relationship with other organizations, such as software development or maintenance. The plan shall be documented and implemented.

NOTE—The plan may be a part of the system configuration management plan.

GUIDANCE:

1—A configuration management plan may also be part of acquisition, supply, development, operation, maintenance plan(s) or any other appropriate plan.

2—Additional guidance on configuration management may be found in ISO 10007.

6.2.2 Configuration identification. This activity consists of the following task:

6.2.2.1 A scheme shall be established for the identification of software configuration items and their versions to be controlled for the project. For each software configuration item and its versions, the following shall be identified: the documentation that establishes the baseline; the version references; and other identification details.

GUIDANCE:

The configuration identification scheme should cover designators for identifying entities to be placed under configuration control and should allow assigning a unique identifier to each software item. The designators should cover the software products to be developed or used and should cover the elements of the software engineering environment. The identification scheme should be at the entity control level, for example, computer files, electronic media, documents, software units, hardware items, and software items. The identification scheme should include the version/revision/release status of each entity.

6.2.3 Configuration control. This activity consists of the following task:

6.2.3.1 The following shall be performed: identification and recording of change requests; analysis and evaluation of the changes; approval or disapproval of the request; and implementation, verification, and release of the modified software item. An audit trail shall exist, whereby each modification, the reason for the modification, and authorization of the modification can be traced. Control and audit of all accesses to the controlled software items that handle safety or security critical functions shall be performed.

GUIDANCE:

1—The principal focus of the requirements in this clause are those items, products, or entities directly associated with the software for the project, i.e., the planning and engineering information in computer files, electronic media, and documents describing the software, and the computer files and electronic media containing the software itself. While other items, products, or entities, such as reports or documents associated with the management and evaluation of those products, are to be controlled at some level, it is not the intention of IEEE/EIA 12207.0 to require all such items, products, or entities to be managed with the same degree of rigor.

2—Configuration control procedures should cover the levels of control through which each identified item, product, or entity is required to pass (for example, author control, project-level control, acquirer control); the persons or groups with the authority to authorize and make changes at each level; and the steps to be followed to request authorization for changes, process change requests, track changes, distribute changes, and preserve past versions. Changes that affect an item, product, or entity already under acquirer control should be proposed to the acquirer in accordance with contractually established forms and procedures, if any.

6.2.4 Configuration status accounting. This activity consists of the following task:

6.2.4.1 Management records and status reports that show the status and history of controlled items including software baseline shall be prepared. Status reports should include the number of changes for a project, latest software item versions, release identifiers, the number of releases, and comparisons of releases.

GUIDANCE:

Records should be prepared and maintained for entities that have been placed under a designated level of configuration control (e.g., project-level or higher configuration control). These records should be maintained for the duration specified in the contract or organizational policy.

6.2.5 Configuration evaluation. This activity consists of the following task:

6.2.5.1 The following shall be determined and ensured: the functional completeness of the software items against their requirements and the physical completeness of the software items (whether their design and code reflect an up-to-date technical description).

6.2.6 Release management and delivery. This activity consists of the following task:

6.2.6.1 The release and delivery of software products and documentation shall be formally controlled. Master copies of code and documentation shall be maintained for the life of the software product. The code and documentation that contain safety or security critical functions shall be handled, stored, packaged, and delivered in accordance with the policies of the organizations involved.

GUIDANCE:

The release management and delivery requirements apply to the party that releases the software product.

6.3 Quality assurance process

The Quality Assurance Process is a process for providing adequate assurance that the software products and processes in the project life cycle conform to their specified requirements and adhere to their established plans. To be unbiased, quality assurance needs to have organizational freedom and authority from persons directly responsible for developing the software product or executing the process in the project. Quality assurance may be internal or external depending on whether evidence of product or process quality is demonstrated to the management of the supplier or the acquirer. Quality assurance may make use of the results of other supporting processes, such as Verification, Validation, Joint Reviews, Audits, and Problem Resolution.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Product assurance;
3) Process assurance;
4) Assurance of quality systems.

6.3.1 Process implementation. This activity consists of the following tasks:

6.3.1.1 A quality assurance process tailored to the project shall be established. The objectives of the quality assurance process shall be to assure that the software products and the processes employed for providing those software products comply with their established requirements and adhere to their established plans.

GUIDANCE:

1—The quality assurance process should be responsible for conducting on-going evaluations of software acquisition, supply, development, maintenance, operation, and supporting process activities and the resulting software products, as applicable, to

a) Assure that each process, activity, and task required by the contract or described in plans is being performed in accordance with the contract and with those plans.

b) Assure that each software product required by a relevant process or contract provisions exists and has undergone software product evaluations, testing, and problem resolution, as required.

2—See annex K of this guide for further guidance on software product evaluations.

6.3.1.2 The quality assurance process should be coordinated with the related Verification (6.4), Validation (6.5), Joint Review (6.6), and Audit (6.7) Processes.

GUIDANCE:

1—Organizations that maintain a quality management system based on ISO 9001 should identify the system's elements that relate directly to the requirements of this clause and the other supporting process clauses and determine how the identified elements comply with these requirements.

2—The quality assurance process should be coordinated with related processes to assure that (1) no unnecessary or harmful duplications in evaluations exist, (2) scheduled evaluations are mutually beneficial and realistic, and (3) conflicting reports from related processes are resolved prior to submission to the problem resolution process.

3—The quality assurance process may use the results of the validation process, the verification process, and other processes and activities to satisfy the quality assurance function.

6.3.1.3 A plan for conducting the quality assurance process activities and tasks shall be developed, documented, implemented, and maintained for the life of the contract. The plan shall include the following:

a) Quality standards, methodologies, procedures, and tools for performing the quality assurance activities (or their references in organization's official documentation);

b) Procedures for contract review and coordination thereof;

c) Procedures for identification, collection, filing, maintenance, and disposition of quality records;

d) Resources, schedule, and responsibilities for conducting the quality assurance activities;

e) Selected activities and tasks from supporting processes, such as Verification (clause 6.4), Validation (6.5), Joint Review (6.6), Audit (6.7), and Problem Resolution (6.8).

GUIDANCE:

For 6.3.1.3 e), identification of the activities and tasks performed by related processes whose reports are to be used for the quality assurance process is intended to facilitate coordination between quality assurance and those related processes.

6.3.1.4 Scheduled and on-going quality assurance activities and tasks shall be executed. When problems or nonconformances with contract requirements are detected, they shall be documented and serve as input to the Problem Resolution Process (6.8). Records of these activities and tasks, their execution, problems, and problem resolutions shall be prepared and maintained.

6.3.1.5 Records of quality assurance activities and tasks shall be made available to the acquirer as specified in the contract.

6.3.1.6 It shall be assured that persons responsible for assuring compliance with the contract requirements have the organizational freedom, resources, and authority to permit objective evaluations and to initiate, effect, resolve, and verify problem resolutions.

GUIDANCE:

Three major requirements exist in 6.3.1.6 with possibly different parties responsible for the different parts: (1) organizational freedom, resources, and authority to permit objective evaluations; (2) responsibility to initiate and effect problem resolution and resolve problems; (3) responsibility to verify problem resolutions. Those responsible for the process or product are responsible for resolving problems found in those processes or products. The party(ies) performing the quality assurance process are responsible for assuring the problem is verified and resolved.

6.3.2 Product assurance. This activity consists of the following tasks:

6.3.2.1 It shall be assured that all the plans required by the contract are documented, comply with the contract, are mutually consistent, and are being executed as required.

6.3.2.2 It shall be assured that software products and related documentation comply with the contract and adhere to the plans.

6.3.2.3 In preparation for the delivery of the software products, it shall be assured that they have fully satisfied their contractual requirements and are acceptable to the acquirer.

GUIDANCE:

1—Software products that have fully satisfied contractual requirements may be assumed to be acceptable to the acquirer.

2—The quality assurance process should be performed to assure that all products specified in the contract exist, have undergone evaluations in accordance with the plans for conducting quality assurance activities and tasks, and satisfy the acceptance criteria in the contract.

3—Product assurance may use the results of verification, validation, and other processes to satisfy product assurance tasks.

6.3.3 Process assurance. This activity consists of the following tasks:

6.3.3.1 It shall be assured that those software life cycle processes (supply, development, operation, maintenance, and supporting processes including quality assurance) employed for the project comply with the contract and adhere to the plans.

6.3.3.2 It shall be assured that the internal software engineering practices, development environment, test environment, and libraries comply with the contract.

6.3.3.3 It shall be assured that applicable prime-contract requirements are passed down to the subcontractor, and that the subcontractor's software products satisfy prime-contract requirements.

6.3.3.4 It shall be assured that the acquirer and other parties are provided the required support and cooperation in accordance with the contract, negotiations, and plans.

6.3.3.5 It should be assured that the software product and process measurements are in accordance with established standards and procedures.

6.3.3.6 It shall be assured that the staff assigned have the skill and knowledge needed to meet the requirements of the project and receive any necessary training.

6.3.4 Assurance of quality systems. This activity consists of the following task:

6.3.4.1 Additional quality management activities shall be assured in accordance with the clauses of ISO 9001.

GUIDANCE:

Application of ISO 9001 should be a part of an organization's overall business activities. The acquirer should not specify use of ISO 9001 as part of a contract. See annex B of this guide for further guidance on compliance. "Additional quality management activities" refers to quality management activities in ISO 9001 that are not specifically covered in IEEE/EIA 12207.0.

6.4 Verification process

The Verification Process is a process for determining whether the software products of an activity fulfill the requirements or conditions imposed on them in the previous activities. For cost and performance effectiveness, verification should be integrated, as early as possible, with the process (such as supply, development, operation, or maintenance) that employs it. This process may include analysis, review and test.

This process may be executed with varying degrees of independence. The degree of independence may range from the same person or different person in the same organization to a person in a different organization with varying degrees of separation. In the case where the process is executed by an organization independent of the supplier, developer, operator, or maintainer, it is called Independent Verification Process.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Verification.

6.4.1 Process implementation. This activity consists of the following tasks:

6.4.1.1 A determination shall be made if the project warrants a verification effort and the degree of organizational independence of that effort needed. The project requirements shall be analyzed for criticality. Criticality may be gauged in terms of:

a) The potential of an undetected error in a system or software requirement for causing death or personal injury, mission failure, or financial or catastrophic equipment loss or damage;

b) The maturity of and risks associated with the software technology to be used;

c) Availability of funds and resources.

6.4.1.2 If the project warrants a verification effort, a verification process shall be established to verify the software product.

6.4.1.3 If the project warrants an independent verification effort, a qualified organization responsible for conducting the verification shall be selected. This organization shall be assured of the independence and authority to perform the verification activities.

6.4.1.4 Based upon the scope, magnitude, complexity, and criticality analysis above, target life cycle activities and software products requiring verification shall be determined. Verification activities and tasks defined in 6.4.2, including associated methods, techniques, and tools for performing the tasks, shall be selected for the target life cycle activities and software products.

6.4.1.5 Based upon the verification tasks as determined, a verification plan shall be developed and documented. The plan shall address the life cycle activities and software products subject to verification, the required verification tasks for each life cycle activity and software product, and related resources, responsibilities, and schedule. The plan shall address procedures for forwarding verification reports to the acquirer and other involved organizations.

6.4.1.6 The verification plan shall be implemented. Problems and nonconformances detected by the verification effort shall be entered into the Problem Resolution Process (6.8). All problems and nonconformances shall be resolved. Results of the verification activities shall be made available to the acquirer and other involved organizations.

6.4.2 Verification. This activity consists of the following tasks:

6.4.2.1 Contract verification. The contract shall be verified considering the criteria listed below:

a) The supplier has the capability to satisfy the requirements.

b) The requirements are consistent and cover user needs.

c) Adequate procedures for handling changes to requirements and escalating problems are stipulated.

d) Procedures and their extent for interface and cooperation among the parties are stipulated, including ownership, warranty, copyright and confidentiality.

e) Acceptance criteria and procedures are stipulated in accordance with requirements.

NOTE—This activity may be used in the contract review (see 6.3.1.3 b).

6.4.2.2 Process verification. The process shall be verified considering the criteria listed below:

a) Project planning requirements are adequate and timely.

b) Processes selected for the project are adequate, implemented, being executed as planned, and compliant with the contract.

c) The standards, procedures, and environments for the project's processes are adequate.

d) The project is staffed and personnel trained as required by the contract.

6.4.2.3 Requirements verification. The requirements shall be verified considering the criteria listed below:

a) The system requirements are consistent, feasible, and testable.

b) The system requirements have been appropriately allocated to hardware items, software items, and manual operations according to design criteria.

c) The software requirements are consistent, feasible, testable, and accurately reflect system requirements.

d) The software requirements related to safety, security, and criticality are correct as shown by suitably rigorous methods.

6.4.2.4 Design verification. The design shall be verified considering the criteria listed below:

a) The design is correct and consistent with and traceable to requirements.

b) The design implements proper sequence of events, inputs, outputs, interfaces, logic flow, allocation of timing and sizing budgets, and error definition, isolation, and recovery.

c) Selected design can be derived from requirements.

d) The design implements safety, security, and other critical requirements correctly as shown by suitably rigorous methods.

6.4.2.5 Code verification. The code shall be verified considering the criteria listed below:

a) The code is traceable to design and requirements, testable, correct, and compliant with requirements and coding standards.

b) The code implements proper event sequence, consistent interfaces, correct data and control flow, completeness, appropriate allocation timing and sizing budgets, and error definition, isolation, and recovery.

c) Selected code can be derived from design or requirements.

d) The code implements safety, security, and other critical requirements correctly as shown by suitably rigorous methods.

6.4.2.6 Integration verification. The integration shall be verified considering the criteria listed below:

a) The software components and units of each software item have been completely and correctly integrated into the software item.

b) The hardware items, software items, and manual operations of the system have been completely and correctly integrated into the system.

c) The integration tasks have been performed in accordance with an integration plan.

6.4.2.7 Documentation verification. The documentation shall be verified considering the criteria listed below:

a) The documentation is adequate, complete, and consistent.
b) Documentation preparation is timely.
c) Configuration management of documents follows specified procedures.

No further guidance provided.

6.5 Validation process

The Validation Process is a process for determining whether the requirements and the final, as-built system or software product fulfills its specific intended use. Validation may be conducted in earlier stages. This process may be conducted as a part of Software Acceptance Support (5.3.13).

This process may be executed with varying degrees of independence. The degree of independence may range from the same person or different person in the same organization to a person in a different organization with varying degrees of separation. In the case where the process is executed by an organization independent of the supplier, developer, operator, or maintainer, it is called Independent Validation Process.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Validation.

6.5.1 Process implementation. This activity consists of the following tasks:

6.5.1.1 A determination shall be made if the project warrants a validation effort and the degree of organizational independence of that effort needed.

6.5.1.2 If the project warrants a validation effort, a validation process shall be established to validate the system or software product. Validation tasks defined below, including associated methods, techniques, and tools for performing the tasks, shall be selected.

6.5.1.3 If the project warrants an independent effort, a qualified organization responsible for conducting the effort shall be selected. The conductor shall be assured of the independence and authority to perform the validation tasks.

6.5.1.4 A validation plan shall be developed and documented. The plan shall include, but is not limited to, the following:

a) Items subject to validation;
b) Validation tasks to be performed;
c) Resources, responsibilities, and schedule for validation;
d) Procedures for forwarding validation reports to the acquirer and other parties.

6.5.1.5 The validation plan shall be implemented. Problems and nonconformances detected by the validation effort shall be entered into the Problem Resolution Process (6.8). All problems and nonconformances shall be resolved. Results of the validation activities shall be made available to the acquirer and other involved organizations.

6.5.2 Validation. This activity shall consist of the following tasks:

6.5.2.1 Prepare selected test requirements, test cases, and test specifications for analyzing test results.

6.5.2.2 Ensure that these test requirements, test cases, and test specifications reflect the particular requirements for the specific intended use.

GUIDANCE:

The “shall” in 6.5.2 applies to 6.5.2.1 and 6.5.2.2. There was an inconsistency in style in the original document and the initial fix was to add the “shall” in 6.5.2. Comments have been submitted to the ISO committee to properly fix the subclauses in the next revision of ISO/IEC 12207.

6.5.2.3 Conduct the tests in subclauses 6.5.2.1 and 6.5.2.2, including:

a) Testing with stress, boundary, and singular inputs;

b) Testing the software product for its ability to isolate and minimize the effect of errors; that is, graceful degradation upon failure, request for operator assistance upon stress, boundary, and singular conditions;

c) Testing that representative users can successfully achieve their intended tasks using the software product.

6.5.2.4 Validate that the software product satisfies its intended use.

6.5.2.5 Test the software product as appropriate in selected areas of the target environment.

NOTE—Other means besides testing (such as, analysis, modeling, simulation, etc.) may be employed for validation.

No further guidance provided.

6.6 Joint review process

The Joint Review Process is a process for evaluating the status and products of an activity of a project as appropriate. Joint reviews are at both project management and technical levels and are held throughout the life of the contract. This process may be employed by any two parties, where one party (reviewing party) reviews another party (reviewed party).

List of activities: This process consists of the following activities:

1) Process implementation;
2) Project management reviews;
3) Technical reviews.

6.6.1 Process implementation. This activity consists of the following tasks:

6.6.1.1 Periodic reviews shall be held at predetermined milestones as specified in the project plan(s). *Ad hoc* reviews should be called when deemed necessary by either party.

6.6.1.2 All resources required to conduct the reviews shall be agreed on by the parties. These resources include personnel, location, facilities, hardware, software, and tools.

6.6.1.3 The parties should agree on the following items at each review: meeting agenda, software products (results of an activity) and problems to be reviewed; scope and procedures; and entry and exit criteria for the review.

GUIDANCE:

Risk items should be included in reviews. See the guidance for 5.1.2.3 and 5.2.4.5 for related information.

6.6.1.4 Problems detected during the reviews shall be recorded and entered into the Problem Resolution Process (6.8) as required.

6.6.1.5 The review results shall be documented and distributed. The reviewing party will acknowledge to the reviewed party the adequacy (for example, approval, disapproval, or contingent approval) of the review results.

6.6.1.6 The parties shall agree on the outcome of the review and any action item responsibilities and closure criteria.

6.6.2 Project management reviews. This activity consists of the following task:

6.6.2.1 Project status shall be evaluated relative to the applicable project plans, schedules, standards, and guidelines. The outcome of the review should be discussed between the two parties and should provide for the following:

a) Making activities progress according to plan, based on an evaluation of the activity or software product status;

b) Maintaining global control of the project through adequate allocation of resources;

c) Changing project direction or determining the need for alternate planning;

d) Evaluating and managing the risk issues that may jeopardize the success of the project.

GUIDANCE:

In addition to contractually required reviews (see guidance for 5.1.2.3 and guidance 2 for 5.2.4.5), the supplier, including the developer, maintainer, or operator, as applicable, may propose additional joint management reviews. The supplier and other applicable parties should plan and take part in such additional reviews at locations and dates proposed by the supplier and approved by the acquirer. Candidate joint management

reviews are identified in annex G of this guide. These reviews should be attended by persons with authority to make cost and schedule decisions and may have the following objectives:

a) Keep management informed about project status, directions being taken, technical agreements reached, and overall status of evolving software products;

b) Resolve issues that could not be resolved at joint technical reviews;

c) Arrive at agreed-upon mitigation strategies for near-term and long-term risks that could not be resolved at joint technical reviews;

d) Identify and resolve management-level issues and risks not raised at joint technical reviews;

e) Obtain commitments and acquirer approvals needed for timely accomplishment of the project.

6.6.3 Technical reviews. This activity consists of the following task:

6.6.3.1 Technical reviews shall be held to evaluate the software products or services under consideration and provide evidence that:

a) They are complete.

b) They comply with their standards and specifications.

c) Changes to them are properly implemented and affect only those areas identified by the Configuration Management Process (6.2).

d) They are adhering to applicable schedules.

e) They are ready for the next planned activity.

f) The development, operation, or maintenance is being conducted according to the plans, schedules, standards, and guidelines of the project.

GUIDANCE:

The supplier, including the developer and/or the maintainer and/or the operator, as applicable, should plan and take part in joint technical reviews at locations and dates proposed by the supplier and approved by the acquirer. These reviews should be attended by persons with technical knowledge of the software products to be reviewed. Support process disciplines (e.g., quality assurance, configuration management, verification, validation) should provide input to or be present at joint reviews. The reviews should focus on in-process and final software products, rather than materials generated especially for the review. The reviews may have the following objectives:

a) Review evolving software products that have met their completion criteria; review and demonstrate proposed technical solutions; provide insight and obtain feedback on the technical effort; and surface and resolve technical issues;

b) Review project status and surface near-term and long-term risks regarding technical, cost, and schedule issues;

c) Arrive at agreed-upon mitigation strategies for identified risks, within the authority of those present;

d) Identify risks and issues to be raised at joint management reviews;

e) Ensure on-going communication between acquirer and supplier technical personnel.

6.7 Audit process

The Audit Process is a process for determining compliance with the requirements, plans, and contract as appropriate. This process may be employed by any two parties, where one party (auditing party) audits the software products or activities of another party (audited party).

List of activities. This process consists of the following activities:

1) Process implementation;
2) Audit.

6.7.1 Process implementation. This activity consists of the following tasks:

6.7.1.1 Audits shall be held at predetermined milestones as specified in the project plan(s).

GUIDANCE:

See the guidance for 5.1.2.3 and 5.2.4.5 for related information.

6.7.1.2 Auditing personnel shall not have any direct responsibility for the software products and activities they audit.

6.7.1.3 All resources required to conduct the audits shall be agreed by the parties. These resources include supporting personnel, location, facilities, hardware, software, and tools.

6.7.1.4 The parties should agree on the following items at each audit: agenda; software products (and results of an activity) to be reviewed; audit scope and procedures; and entry and exit criteria for the audit.

6.7.1.5 Problems detected during the audits shall be recorded and entered into the Problem Resolution Process (6.8) as required.

6.7.1.6 After completing an audit, the audit results shall be documented and provided to the audited party. The audited party shall acknowledge to the auditing party any problems found in the audit and related problem resolutions planned.

6.7.1.7 The parties shall agree on the outcome of the audit and any action item responsibilities and closure criteria.

6.7.2 Audit. This activity consists of the following task:

6.7.2.1 Audits shall be conducted to ensure that:

a) As-coded software products (such as a software item) reflect the design documentation.

b) The acceptance review and testing requirements prescribed by the documentation are adequate for the acceptance of the software products.

c) Test data comply with the specification.

d) Software products were successfully tested and meet their specifications.

e) Test reports are correct and discrepancies between actual and expected results have been resolved.

f) User documentation complies with standards as specified.

g) Activities have been conducted according to applicable requirements, plans, and contract.

h) The costs and schedules adhere to the established plans.

No further guidance provided.

6.8 Problem resolution process

The Problem Resolution Process is a process for analyzing and resolving the problems (including nonconformances), whatever their nature or source, that are discovered during the execution of development, operation, maintenance, or other processes. The objective is to provide a timely, responsible, and documented means to ensure that all discovered problems are analyzed and resolved and trends are recognized.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Problem resolution.

6.8.1 Process implementation. This activity consists of the following task:

6.8.1.1 A problem resolution process shall be established for handling all problems (including nonconformances) detected in the software products and activities. The process shall comply with the following requirements:

a) The process shall be closed-loop, ensuring that: all detected problems are promptly reported and entered into the Problem Resolution Process; action is initiated on them; relevant parties are advised of the existence of the problem as appropriate; causes are identified, analyzed, and, where possible, eliminated; resolution and disposition are achieved; status is tracked and reported; and records of the problems are maintained as stipulated in the contract.

b) The process should contain a scheme for categorizing and prioritizing the problems. Each problem should be classified by the category and priority to facilitate trend analysis and problem resolution.

c) Analysis shall be performed to detect trends in the problems reported.

d) Problem resolutions and dispositions shall be evaluated: to evaluate that problems have been resolved, adverse trends have been reversed, and changes have been correctly implemented in the appropriate software products and activities; and to determine whether additional problems have been introduced.

GUIDANCE:

Problems found in products under author control should be resolved by the author. Problems found in products not under author control should be treated by the problem resolution process. See guidance 2 in 5.3.1.2 for further information on levels of control. See annex J of this guide for a sample scheme for prioritizing and categorizing problems.

6.8.2 Problem resolution. This activity consists of the following task:

6.8.2.1 When problems (including nonconformances) have been detected in a software product or an activity, a problem report shall be prepared to describe each problem detected. The problem report shall be used as part of the closed-loop process described above: from detection of the problem, through investigation, analysis and resolution of the problem and its cause, and onto trend detection across problems.

No further guidance provided.

7 Organizational life cycle processes

This clause defines the following organizational life cycle processes:

1) Management process;
2) Infrastructure process;
3) Improvement process;
4) Training process.

The activities and tasks in an organizational process are the responsibility of the organization using that process. The organization ensures that the process is in existence and functional.

No further guidance provided.

7.1 Management process

The Management Process contains the generic activities and tasks, which may be employed by any party that has to manage its respective process(es). The manager is responsible for product management, project management, and task management of the applicable process(es), such as the acquisition, supply, development, operation, maintenance, or supporting process.

List of activities: This process consists of the following activities:

1) Initiation and scope definition;
2) Planning;
3) Execution and control;
4) Review and evaluation;
5) Closure.

7.1.1 Initiation and scope definition. This activity consists of the following tasks:

7.1.1.1 The management process shall be initiated by establishing the requirements of the process to be undertaken.

7.1.1.2 Once the requirements are established, the manager shall establish the feasibility of the process by checking that the resources (personnel, materials, technology, and environment) required to execute and manage the process are available, adequate, and appropriate and that the time-scales to completion are achievable.

7.1.1.3 As necessary, and by agreement of all parties concerned, the requirements of the process may be modified at this point to achieve the completion criteria.

7.1.2 Planning. This activity consists of the following task:

7.1.2.1 The manager shall prepare the plans for execution of the process. The plans associated with the execution of the process shall contain descriptions of the associated activities and tasks and identification of the software products that will be provided. These plans shall include, but are not limited to, the following:

a) Schedules for the timely completion of tasks;
b) Estimation of effort;
c) Adequate resources needed to execute the tasks;
d) Allocation of tasks;
e) Assignment of responsibilities;
f) Quantification of risks associated with the tasks or the process itself;
g) Quality control measures to be employed throughout the process;
h) Costs associated with the process execution;
i) Provision of environment and infrastructure.

GUIDANCE:

If all of the information is provided in other plans, the manager is not required to produce additional plans. See the guidance for 5.2.4.5 for related information.

7.1.3 Execution and control. This activity consists of the following tasks:

7.1.3.1 The manager shall initiate the implementation of the plan to satisfy the objectives and criteria set, exercising control over the process.

7.1.3.2 The manager shall monitor the execution of the process, providing both internal reporting of the process progress and external reporting to the acquirer as defined in the contract.

7.1.3.3 The manager shall investigate, analyze, and resolve the problems discovered during the execution of the process. The resolution of problems may result in changes to plans. It is the manager's responsibility to ensure the impact of any changes is determined, controlled, and monitored. Problems and their resolution shall be documented.

7.1.3.4 The manager shall report, at agreed points, the progress of the process, declaring adherence to the plans and resolving instances of the lack of progress. These include internal and external reporting as required by the organizational procedures and the contract.

7.1.4 Review and evaluation. This activity consists of the following tasks:

7.1.4.1 The manager shall ensure that the software products and plans are evaluated for satisfaction of requirements.

7.1.4.2 The manager shall assess the evaluation results of the software products, activities, and tasks completed during the execution of the process for achievement of the objectives and completion of the plans.

GUIDANCE:

Annex H of this guide contains suggested software measurement categories that may be used in evaluations.

7.1.5 Closure. This activity consists of the following tasks:

7.1.5.1 When all software products, activities, and tasks are completed, the manager shall determine whether the process is complete taking into account the criteria as specified in the contract or as part of organization's procedure.

7.1.5.2 The manager shall check the results and records of the software products, activities, and tasks employed for completeness. These results and records shall be archived in a suitable environment as specified in the contract.

No further guidance provided.

7.2 Infrastructure process

The Infrastructure Process is a process to establish and maintain the infrastructure needed for any other process. The infrastructure may include hardware, software, tools, techniques, standards, and facilities for development, operation, or maintenance.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Establishment of the infrastructure;
3) Maintenance of the infrastructure.

7.2.1 Process implementation. This activity consists of the following tasks:

7.2.1.1 The infrastructure should be defined and documented to meet the requirements of the process employing this process, considering the applicable procedures, standards, tools, and techniques.

7.2.1.2 The establishment of the infrastructure should be planned and documented.

7.2.2 Establishment of the infrastructure. This activity consists of the following tasks:

7.2.2.1 The configuration of the infrastructure should be planned and documented. Functionality, performance, safety, security, availability, space requirements, equipment, costs, and time constraints should be considered.

7.2.2.2 The infrastructure shall be installed in time for execution of the relevant process.

7.2.3 Maintenance of the infrastructure. This activity consists of the following task:

7.2.3.1 The infrastructure shall be maintained, monitored, and modified as necessary to ensure that it continues to satisfy the requirements of the process employing this process. As part of maintaining the infrastructure, the extent to which the infrastructure is under configuration management shall be defined.

No further guidance provided.

7.3 Improvement process

The Improvement Process is a process for establishing, assessing, measuring, controlling, and improving a software life cycle process.

List of activities. This process consists of the following activities:

1) Process establishment;
2) Process assessment;
3) Process improvement.

7.3.1 Process establishment. This activity consists of the following task:

7.3.1.1 The organization shall establish a suite of organizational processes for all software life cycle processes as they apply to its business activities. The processes and their application to specific cases shall be documented in organization's publications. As appropriate, a process control mechanism should be established to develop, monitor, control, and improve the process(es).

7.3.2 Process assessment. This activity consists of the following tasks:

7.3.2.1 A process assessment procedure should be developed, documented, and applied. Assessment records should be kept and maintained.

7.3.2.2 The organization shall plan and carry out reviews of the process at appropriate intervals to ensure its continuing suitability and effectiveness in the light of assessment results.

7.3.3 Process improvement. This activity consists of the following tasks:

7.3.3.1 The organization shall effect such improvements to its processes as it determines to be necessary as a result of process assessment and review. Process documentation should be updated to reflect improvement in the organizational processes.

7.3.3.2 Historical, technical, and evaluation data should be collected and analyzed to gain an understanding of the strengths and weaknesses of the employed processes. These analyses should be used as feedback to improve these processes, to recommend changes in the direction of the projects (or subsequent projects), and to determine technology advancement needs.

7.3.3.3 Quality cost data should be collected, maintained, and used to improve the organization's processes as a management activity. These data shall serve the purpose of establishing the cost of both the prevention and resolution of problems and non-conformity in software products and services.

No further guidance provided.

7.4 Training process

The Training Process is a process for providing and maintaining trained personnel. The acquisition, supply, development, operation, or maintenance of software products is largely dependent upon knowledgeable and skilled personnel. For example: developer personnel should have essential training in software management and software engineering. It is, therefore, imperative that personnel training be planned and implemented early so that trained personnel are available as the software product is acquired, supplied, developed, operated, or maintained.

List of activities. This process consists of the following activities:

1) Process implementation;
2) Training material development;
3) Training plan implementation.

7.4.1 Process implementation. This activity consists of the following task:

7.4.1.1 A review of the project requirements shall be conducted to establish and make timely provision for acquiring or developing the resources and skills required by the management and technical staff. The types and levels of training and categories of personnel needing training shall be determined. A training plan, addressing implementation schedules, resource requirements, and training needs, should be developed and documented.

7.4.2 Training material development. This activity consists of the following task:

7.4.2.1 Training manuals, including presentation materials used in providing training, should be developed.

7.4.3 Training plan implementation. This activity consists of the following tasks:

7.4.3.1 The training plan shall be implemented to provide training to personnel. Training records should be maintained.

7.4.3.2 It should be ensured that the right mix and categories of appropriately trained personnel are available for the planned activities and tasks in a timely manner.

No further guidance provided.

Annex A

IEEE/EIA 12207.0 Annex A
(normative)
Tailoring process

The Tailoring Process is a process for performing basic tailoring of this International Standard for a software project. This annex provides requirements for tailoring this International Standard.

List of activities. This process consists of the following activities:

1) Identifying project environment;
2) Soliciting inputs;
3) Selecting processes, activities, and tasks;
4) Documenting tailoring decisions and rationale.

A.1 Identifying project environment. This activity consists of the following task:

A.1.1 Characteristics of the project environment that are going to influence tailoring shall be identified. Some of the characteristics may be: life cycle model; current system life cycle activity; system and software requirements; organizational policies, procedures and strategies; size, criticality and types of the system, software product or service; and number of personnel and parties involved.

A.2 Soliciting inputs. This activity consists of the following task:

A.2.1 Inputs from the organizations that will be affected by tailoring decisions shall be solicited. Users, support personnel, contracting officers, potential bidders should be involved in tailoring.

A.3 Selecting processes, activities, and tasks. This activity consists of the following tasks:

A.3.1 The processes, activities, and tasks that are to be performed shall be decided. These include the documentation to be developed and who are to be responsible for them. For this purpose, this International Standard should be evaluated against relevant data gathered in clauses A.1 and A.2.

A.3.2 The processes, activities, and tasks that were decided upon in A.3.1 but not provided in this International Standard shall be specified in the contract itself. Organizational life cycle processes (clause 7) should be evaluated to determine whether they could provide for these processes, activities, and tasks.

A.3.3 In this International Standard, requirements are indicated by tasks that contain “shall” or “will.” These tasks should be carefully considered for whether they should be kept or deleted for a given project or a given business sector. Factors to be considered include but are not limited to: risk, cost, schedule, performance, size, criticality, and human interface.

A.4 Documenting tailoring decisions and rationale. This activity consists of the following task:

A.4.1 All tailoring decisions shall be documented together with the rationale for the decisions.

GUIDANCE:

1—Annex A of IEEE/EIA 12207.0 was originally written (in ISO/IEC 12207) for the purpose of tailoring a project. Annex F of IEEE/EIA 12207.0 (annex B of this guide) expands the treatment of compliance to deal with processes at the organizational level.

2—In order to claim compliance with IEEE/EIA 12207.0, a project may simply reference the organization’s claim of compliance and document the project-specific interpretation of any clauses of IEEE/EIA 12207.0 that reference “the contract.”

3—In some cases, it may be necessary to perform additional tailoring because the contracted scope of activity may not cover all of the processes, activities, and tasks of IEEE/EIA 12207.0. This tailoring may be documented by simply listing the processes, activities, and tasks that are out of scope. A description of the rationale for tailoring is also recommended.

4—Tailoring is not intended as license to modify the requirements of the standard. Tailoring has two purposes: (1) to delete those provisions of the standard that fall outside of the contracted scope of the work and (2) to interpret the provisions of the standard that make reference to “the contract.” The guidance provided in annex B of this guide may be helpful.

Annex B

IEEE/EIA 12207.0 Annex F
(normative)
Compliance

In accordance with clause 1.4, paragraph 2, of the International Standard, the U.S. defines compliance with this U.S. standard as explained in this annex.

F.1 Definition of compliance

Compliance with this U.S. standard is defined similarly to clause 1.4, paragraph 1, of the International Standard:

> Compliance with this Standard is defined as the performance of all the processes, activities, and tasks selected from this Standard in the Tailoring Process (annex A) for the software project. The performance of a process or an activity is complete when all its required tasks are performed in accordance with the pre-established criteria and the requirements specified in the contract as applicable.

The overall interpretation of compliance may vary in different situations, as explained in clause F.2. The set of tasks “selected ... in the Tailoring Process” may be constrained, as explained in clause F.3. The performance of the processes, activities, and tasks “in accordance with the pre-established criteria” may be achieved in a number of ways, as explained in clause F.4.

F.2 Compliance situations

Compliance may be interpreted differently for various situations. The relevant situation shall be identified in the claim of compliance:

a) When compliance is claimed for an organization, the organization shall make public a document declaring its tailoring of the processes, activities, and tasks and its interpretation of any clauses of the standard that reference “the contract.”

   NOTE—One possible way for an organization to deal with clauses that cite “the contract” is to specify that they shall be interpreted in the project plans for any particular project.

b) When compliance is claimed for a project, the project plans or the contract shall document the tailoring of the processes, activities and tasks and the interpretation of any clauses of the standard that reference “the contract.”

   NOTE—A project’s claim of compliance is typically specified with respect to the organization’s claim of compliance.

c) When compliance is claimed for a multi-supplier program, it may be the case that no individual project can claim compliance because no single contract calls for the performance of all required processes and activities. Nevertheless, the program, as a whole, may claim compliance if each of the required processes and activities is performed by an identified party. The program plans shall document the tailoring of the processes, activities and tasks, and their assignment to the various parties, as well as the interpretation of any clauses of the standard that reference “the contract.”

d) When the standard is used as a basis for regulatory decisions, supplemental standards, legal regulations or sector-specific requirements may constrain the tailoring and the interpretation of the clauses of the standard that reference “the contract.” In this case, the project plans shall specify the tailoring and the interpretation of the clauses of the standard that reference “the contract.”

Annex B *(Continued)*

F.3 Level of compliance

One of the following levels of compliance shall be asserted. The selected level shall be identified in the claim of compliance:

a) Tailored: The minimum set of required processes, activities, and tasks is determined by tailoring in accordance with annex A.

b) Absolute: The minimum set of required processes, activities, and tasks are all of those specified as mandatory (i.e., clauses containing "shall" or "will") in the text of the standard.

NOTE—Absolute compliance may be claimed for selected processes even if absolute compliance with the entire standard is not claimed.

F.4 Compliance criteria

Performance of any process of the standard shall be completed using either of two sets of criteria. For each process, the claim of compliance shall cite which of the sets has been chosen:

a) Accomplishment of all of the requirements of the activities and tasks of the process as specified in clauses 5, 6, and 7 of IEEE/EIA 12207, in accordance with the selected level of compliance and the chosen tailoring.

b) Accomplishment of the process objectives via an "alternative method": To be acceptable, alternative methods must accomplish the process objectives specified in annex G of IEEE/EIA 12207, must accomplish the life cycle data objectives specified in annex H of IEEE/EIA 12207, must not hamper compliance of any other process, and must be specified by plans, standards or other documents. The alternative method and the means of compliance shall be specified or cited in the claim of compliance.

NOTE—Possible "alternative methods" for some processes are specified in IEEE/EIA 12207.2.

GUIDANCE:

1—The U.S. adaptation of ISO/IEC 12207 has directed the emphasis of the standard in two important ways:

- Toward the organization or the enterprise as the complying entity;
- Toward a more constrained concept of tailoring.

This annex explains some of the ramifications of these decisions and how they affect compliance.

2—ISO/IEC 12207 addresses the individual project only in its compliance clause. The method used to tailor the standard to the needs of the project is to simply delete any portion of the standard thought to be unsuitable to the particular project.

3—IEEE/EIA 12207.0 addresses three additional compliance situations: the entire organization or enterprise, the multi-supplier program, and regulatory situations. Organizational compliance will be discussed first.

4—The most highly desired application of IEEE/EIA 12207.0 is for the adopting organization to develop and implement policies and procedures that apply to all of the projects to be conducted by the organization and comply with the requirements of IEEE/EIA 12207.0 at the absolute level of compliance. This approach means that the organizational processes would be capable of executing every one of the processes, activities and tasks indicated as mandatory in the body of the standard. It is permissible for an organization to implement only some of the required processes—for example, it might omit the Acquisition process if it never contracts for the development of software. The standard permits the organization to claim absolute compliance for each specific implemented process. Less highly desired, but still complying, would be the situation where one or more of the organization's processes omit certain of the required activities and tasks specified in the standard. In this case, the level of compliance for those processes would be "tailored" rather than "absolute" and the organization's claim of compliance would have to reference a document that describes the tailoring. In exceptional cases, the

organization may desire to implement a process that departs broadly from the requirements of the standard. Such processes may be claimed as complying if they satisfy the process and data objectives as specified in IEEE/EIA 12207.0 F.4 (b).

5—One additional issue must be addressed in organizational compliance: the question of how to interpret the clauses of the standard that reference “the contract.” One possible solution is for the organizational processes to specify that such clauses must be interpreted by each project that applies the organizational processes and that the interpretation of the clauses would be described in the project’s claim of compliance.

6—Within an organization that has implemented complying processes, any particular project should adopt and apply the organizational processes for use on the project. Wholesale tailoring of the organizational processes is unnecessary and unwise. The project planners should designate which of the organizational processes, activities, and tasks are within the scope of the contract. For example, if the contract is solely for software *development*, then the organizational *maintenance* and *operation* processes would be designated as inapplicable. In some cases, it may be necessary to determine applicability at the activity or even at the task levels. In any case, the project’s claim of compliance would reference the organizational processes and enumerate the portions regarded as inapplicable. The project’s claim of compliance would also describe the interpretation to be applied to each of the clauses of the standard that reference “the contract.”

7—With this explanation, it can be seen that the concept called “tailoring” in ISO/IEC 12207 is intended to be applied in a constrained and disciplined fashion in IEEE/EIA 12207.0. In IEEE/EIA 12207.0, the relevant concepts are to *select* the applicable portions of the standard and to *parameterize* the standard based upon specific requirements. In no case should one consider tailoring as license to modify the requirements of the standard.

8—IEEE/EIA 12207.0 also permits a multi-supplier program to claim compliance with the standard. The typical situation here is that an organization is developing a software system but has chosen to subcontract some of the labor to different contractors. The organization is actively and directly managing the subcontracted labor. In this situation, it might be the case that no single organization is itself executing all of the requirements of an entire process; hence, no single organization could claim compliance to the standard. Nevertheless, if all of the requirements are being satisfied by each organization performing relevant activities or tasks, then the overall *program* could claim compliance even if no single organization could claim it.

9—Finally, specific provisions are made for regulatory situations. It is envisioned that the regulating entity would adopt IEEE/EIA 12207.0 for the implementation of processes to be used by the regulated organization. The point of the provision is simply to recognize that the organizational processes to be applied by regulated parties may come from the regulating organization.

Annex C

IEEE/EIA 12207.0 Annex G
(normative)
Life cycle processes objectives

The purpose of this annex is to describe the basic objectives to be considered in meeting the intent of each life cycle process defined in ISO/IEC 12207. For this annex, “customer” is defined to be the recipient of a product provided by a suppler [sic]. In the contractual situation, the customer is called the “acquirer.” The “customer” may be the ultimate consumer, user, beneficiary, or purchaser. The “customer” can be external or internal to the organization. Data engineering activities are assumed to be covered under the heading of the analogous software engineering activities (e.g., requirements and design).

The objectives presented below are intended to be complete. For a particular project, only the subset of those objectives that conform to the tailoring of the standard would be applicable.

G.1 Acquisition process

The use of the Acquisition process should achieve the following objectives:

a) Develop a contract, including tailoring of the standard, that clearly expresses the expectation, responsibilities, and liabilities of both the acquirer and the supplier;

b) Obtain products and/or services that satisfy the customer need;

c) Manage the acquisition so that specified constraints (e.g., cost, schedule and quality) and goals (e.g., degree of software reuse) are met;

d) Establish a statement of work to be performed under contract;

e) Qualify potential suppliers through an assessment of their capability to perform the required software;

f) Select qualified suppliers to perform defined portions of the contract;

g) Establish and manage commitments to and from the supplier;

h) Regularly exchange progress information with the supplier;

i) Assess compliance of the supplier against the agreed upon plans, standards and procedures;

j) Assess the quality of the supplier’s delivered products and services;

k) Establish and execute acceptance strategy and conditions (criteria) for the software product or service being acquired;

l) Establish a means by which the acquirer will assume responsibility for the acquired software product or service.

G.2 Audit process

The use of the Audit process should achieve the following objectives:

a) Determine compliance with requirements, plans, and contract, as appropriate;

b) Arrange the conduct of audits of work products or process performance by a qualified independent party, as specified in the plans;

c) Conduct follow-up audits to assess corrective action(s), closure, and root cause actions.

Annex C *(Continued)*

G.3 Configuration Management process

The use of the Configuration Management process should achieve the following objectives:

a) Identify, define, and control all relevant items of the project;

b) Control modifications of the items;

c) Record and report the status of items and modification requests;

d) Ensure the completeness of the items;

e) Control storage, handling, release, and delivery of the items.

G.4 Development process

The use of the Development process should achieve the following objectives:

a) Develop requirements of the system that match the customer's stated and implied needs;

b) Propose an effective solution that identifies the main elements of the system;

c) Allocate the defined requirements to each of those main elements;

d) Develop a system release strategy;

e) Communicate the requirements, proposed solution and their relationships to all affected parties;

f) Define the requirements allocated to software components of the system and their interfaces to match the customer's stated and implied needs;

g) Develop software requirements that are analyzed, correct, and testable;

h) Understand the impact of software requirements on the operating environment;

i) Develop a software release strategy;

j) Approve and update the software requirements, as needed;

k) Communicate the software requirements to all affected parties;

l) Develop an architectural design;

m) Define internal and external interfaces of each software component;

n) Establish traceability between system requirements and design and software requirements, between software requirements and software design, and between software requirements and tests;

o) Define verification criteria for all software units against the software requirements;

p) Produce software units defined by the design;

q) Accomplish verification of the software units against the design;

r) Develop an integration strategy for software units consistent with the release strategy;

s) Develop acceptance criteria for software unit aggregates that verify compliance with the software requirements allocated to the units;

t) Verify software aggregates using the defined acceptance criteria;

u) Verify integrated software using the defined acceptance criteria;

v) Record the results of the software tests;

w) Develop a regression strategy for retesting aggregates, or the integrated software, should a change in components be made;

x) Develop an integration plan to build system unit aggregates according to the release strategy;

y) Define acceptance criteria for each aggregate to verify compliance with the system requirements allocated to the units;

Annex C *(Continued)*

z) Verify system aggregates using the defined acceptance criteria;

aa) Construct an integrated system demonstrating compliance with the system requirements (functional, nonfunctional, operations and maintenance);

ab) Record the results of the system tests;

ac) Develop a regression strategy for retesting aggregates or the integrated system should a change in components be made;

ad) Identify transition concerns, such as availability or [sic] work products, availability of system resources to resolve problems and adequately test before fielding corrections, maintainability, and assessment of transitioned work products.

G.5 Documentation process

The use of the Documentation process should achieve the following objectives:

a) Identify all documents to be produced by the process or project;

b) Specify the content and purpose of all documents and plan and schedule their production;

c) Identify the standards to be applied for development of documents;

d) Develop and publish all documents in accordance with identified standards and in accordance with nominated plans;

e) Maintain all documents in accordance with specified criteria.

NOTE—Refer to clause H.5 of [IEEE/EIA 12207.0] for presentation form of software life cycle data.

G.6 Improvement process

The use of the Improvement process should achieve the following objectives:

a) Establish a well-defined and maintained standard set of processes, along with a description of the applicability of each process;

b) Identify the detailed tasks, activities, and associated work products for each standard process, together with expected criteria;

c) Establish a deployed specific process for each project, tailored from the standard process in accordance with the needs of the project;

d) Establish and maintain information and data related to the use of the standard process for specific projects;

e) Understand the relative strengths and weaknesses of the organization’s standard software processes;

f) Make changes to standard and defined processes in a controlled way;

g) Implement planned and monitored software process improvement activities in a coordinated manner across the organization.

G.7 Infrastructure process

The use of the Infrastructure process should achieve the following objectives:

a) Establish and maintain a well-defined software engineering environment, consistent with and supportive of the set of standard processes and organizational methods and techniques;

b) Tailor the software engineering environment to the needs of the project and the project team;

c) Develop a software engineering environment that supports project team members regardless of the performance location of process activities;

d) Implement a defined and deployed strategy for reuse.

Annex C *(Continued)*

G.8 Joint Review process

The use of the Joint Review process should achieve the following objectives:

a) Evaluate the status and products of an activity of a process through joint review activities between the parties to a contract;

b) Establish mechanisms to ensure that action items raised are recorded for action.

G.9 Maintenance process

The use of the Maintenance process should achieve the following objectives:

a) Define the impact of organization, operations, and interfaces on the existing system in operation;

b) Identify and update appropriate life cycle data;

c) Develop modified system components with associated documentation and tests that demonstrate that the system requirements are not compromised;

d) Migrate system and software upgrades to the user's environment;

e) Ensure fielding of new systems or versions does not adversely affect ongoing operations;

f) Maintain the capability to resume processing with prior versions.

G.10 Management process

The use of the Management process should achieve the following objectives:

a) Define the scope of the work for the project;

b) Identify, size, estimate, plan, track, and measure the tasks and resources necessary to complete the work;

c) Identify and manage interfaces between elements in the project and with other projects and organizational units;

d) Take corrective action when project targets are not achieved;

e) Establish quality goals, based on the customer's quality requirements, for various checkpoints within the project's software life cycle;

f) Establish product performance (memory, processing, communications) goals, based on the customer's requirements, for various checkpoints within the project's software life cycle;

g) Define and use metrics that measure the results of project activities or tasks, at checkpoints within the project's life cycle, to assess whether the technical, quality, and product performance goals have been achieved;

h) Establish criteria, metrics, and procedures for identifying software engineering practices and integrate improved practices into the appropriate software life cycle processes and methods;

i) Perform the identified quality activities and confirm their performance;

j) Take corrective action when technical, quality, and product performance goals are not achieved;

k) Determine the scope of risk management to be performed for the project;

l) Identify risks to the project as they develop;

m) Analyze risks and determine the priority in which to apply resources to mitigate those risks;

n) Define, implement, and assess appropriate risk mitigation strategies;

o) Define, apply, and assess risk metrics to measure the change in the risk state and the progress of the mitigation activities;

p) Establish an environment that supports effective interaction between individuals and groups;

q) Take corrective action when expected progress is not achieved.

Annex C *(Continued)*

G.11 Operation process

The use of the Operation process should achieve the following objectives:

a) Identify and mitigate operational risks for the software introduction and operation;

b) Operate the software in its intended environment according to documented procedures;

c) Provide operational support by resolving operational problems and handling user inquires and requests;

d) Provide assurance that software (and host system) capacities are adequate to meet user needs;

e) Identify customer support service needs on an ongoing basis;

f) Assess customer satisfaction with both the support services being provided and the product itself on an ongoing basis;

g) Deliver needed customer services.

G.12 Problem Resolution process

The use of the Problem Resolution process should achieve the following objectives:

a) Provide a timely, responsive, and documented means to ensure that all discovered problems are analyzed and resolved;

b) Provide a mechanism for recognizing and acting on trends in problems identified.

G.13 Quality Assurance process

The use of the Quality Assurance process should achieve the following objectives:

a) Identify, plan, and schedule quality assurance activities for the process or product;

b) Identify quality standards, methodologies, procedures, and tools for performing quality assurance activities and tailor to the project;

c) Identify resources and responsibilities for the performance of quality assurance activities;

d) Establish and guarantee the independence of those responsible for performing quality assurance activities;

e) Perform the identified quality assurance activities in line with the relevant plans, procedures, and schedules;

f) Apply organizational quality management systems to the project.

G.14 Supply process

The use of the Supply process should achieve the following objectives:

a) Establish clear and ongoing communication with the customer;

b) Define documented and agreed customer requirements, with managed changes;

c) Establish a mechanism for ongoing monitoring of customer needs;

d) Establish a mechanism for ensuring that customers can easily determine the status and disposition of their requests;

e) Determine requirements for replication, distribution, installation, and testing of the system containing software or stand-alone software product;

f) Package the system containing software or the stand-alone software product in a way that facilitates its efficient and effective replication, distribution, installation, testing, and operation;

g) Deliver a quality system containing software or stand-alone software product to the customer, as defined by the requirements, and install in accordance with the identified requirements.

Annex C *(Continued)*

G.15 Training process

The use of the Training process should achieve the following objectives:

a) Identify the roles and skills required for the operations of the organization and the project;

b) Establish formal procedures by which talent is recruited, selected, and transitioned into assignments in the organization;

c) Design and conduct training to ensure that all individuals have the skills required to perform their assignments;

d) Identify and recruit or train, as appropriate, individuals with the required skills and competencies to perform the organizational and project roles;

e) Establish a work force with the skills to share information and coordinate their activities efficiently;

f) Define objective criteria against which unit and individual training performance can be measured, to provide performance feedback, and to enhance performance continuously.

G.16 Validation process

The use of the Validation process should achieve the following objectives:

a) Identify criteria for validation of all required work products;

b) Perform required validation activities;

c) Provide evidence that the work products, as developed, are suitable for their intended use.

G.17 Verification process

The use of the Verification process should achieve the following objectives:

a) Identify criteria for verification of all required work products;

b) Perform requirements verification activities;

c) Find and remove defects from products produced by the project.

GUIDANCE:

Further guidance on the use of life cycle processes objectives may be found in IEEE/EIA 12207.0 F.4 b) and in the guidance for annex B of this guide.

Annex D

IEEE/EIA 12207.0 Annex H
(normative)
Life cycle data objectives

The purpose of this annex is to describe the basic principles to be considered in preparing data during the execution of the software life cycle processes of IEEE/EIA 12207.

H.1 Purpose of software life cycle data

The life cycle data should support the following actions:

a) Describe and record information about a software product during its life cycle;

b) Assist usability and maintainability of a software product;

c) Define and control life cycle processes;

d) Communicate information about the system, software product or service, and project to those who need it;

e) Provide a history of what happened during development and maintenance to support management and process improvement;

f) Provide evidence that the processes were followed.

H.2 Operations on software life cycle data

The life cycle data should be supported by the following operations:

a) Create;

b) Read;

c) Update;

d) Delete.

H.3 Characteristics of software life cycle data

The life cycle data should adhere to the following characteristics:

a) Unambiguous: Information is unambiguous if it is described in terms that only allow a single interpretation, aided, if necessary, by a definition.

b) Complete: Information is complete if it includes necessary, relevant requirements and/or descriptive material, responses are defined for the range of valid input data, and terms and units of measure are defined.

c) Verifiable: Information is verifiable if it can be checked for correctness by a person or tool.

d) Consistent: Information is consistent if there are no conflicts within it.

e) Modifiable: Information is modifiable if it is structured and has a style such that changes can be made completely, consistently, and correctly while retaining the structure.

f) Traceable: Information is traceable if the origin of its components can be determined.

g) Presentable: Information is presentable if it can be retrieved and viewed.

H.4 Basic types of software life cycle data

The life cycle data should contain content in the following areas:

a) Requirements data: Expected functionality, operational context, performance constraints and expectations, basis for qualification testing, and key decision rationale.

Annex D *(Continued)*

b) Design data: Architecture, algorithms, design constraints, mapping to requirements, and key decision rationale.

c) Test data: Test strategy and criteria, cases (what to test), procedures (how to carry out tests), test results, and key decision rationale.

d) Configuration data: Configuration description, build instructions, reference to source code, reference to object code, data integrity approach, description of development environment, and key decision rationale.

e) User data: Software overview, system access information, commands and responses, error messages, operational environment, and key decision rationale.

f) Management data: Management plans, status reports, management indicators, criteria and key decision rationale, and contract and other procurement information.

g) Quality data: Quality plans and procedures, corrective action status, root cause analysis, product quality characteristics and process measurement data, and criteria and key decision rationale.

H.5 Presentation form of software life cycle data

The presentation form of life cycle data should

a) Be appropriate to support the purpose of the life cycle data;

b) Support the retrieval and review of data of a software item during its life cycle;

c) Support the basic operations on data of a software item during its life cycle;

d) Be selected subject to concurrence of the users of the data.

NOTE—In preparing or finalizing the contract, the acquirer should specify the requirements for data delivery, taking into account the maintenance strategy. Some of the choices are

1) Raw data—Repositories of the development tools such as CASE tools, databases, file systems, and other tool repositories.

2) On-line publishing systems—Data assembled and formatted for presentation by systems such as: word processors, World Wide Web publishing and display systems, SGML viewers.

3) Hard copy print—Traditional paper document form.

GUIDANCE:

Further guidance on life cycle data is available in IEEE/EIA 12207.1.

Annex E

IEEE/EIA 12207.0 Annex J
(normative)
Errata

This annex identifies those clauses in ISO/IEC 12207 that have ambiguities that could not be removed prior to its publication. They are presented here to minimize any possible misinterpretation or misapplication of this standard. Any suggested change is indicated in *italics*. All clauses and paragraphs refer to ISO/IEC 12207.

These identified clauses will be forwarded to the ISO/IEC JTC1 for appropriate action.

1. Clause 1.2, paragraph 4, is clarified below.

   This clause does not imply that the suppliers or developers of off-the-shelf software should not use ISO/IEC 12207 when developing, operating, or maintaining such software.

2. Clause 1.5, paragraph 6 should read as follows:

   In this International Standard, there are a number of lists for tasks; none of these is presumed to be exhaustive — they are intended as examples *unless introduced by a clause containing a "shall" or a "will."*

3. Clause 5.1.3.5, sentence 2. "Shall" should be changed to "will"

4. Add the following clause to 5.3.1.2:

   e) Establish baselines for each configuration item at appropriate times, as determined by the acquirer and the supplier.

5. Delete sentence 2 of clause 5.3.4.3.

6. Delete clause 5.3.9.5.b.

7. Delete clause 5.3.11.4.b.

8. Clause 6.1, Preamble. The following should be added as a second paragraph to the preamble:

   Execution of this process by an organization results in the establishment of internal documentation standards (such as standards for program management plan and software design document) in a suitable media — paper, electronic, or other. The terms used in this process need to be interpreted accordingly for a given media or domain.

9. Clause 6.2, preamble, line 2. "Baseline" should be deleted. The resulting sentence should read as follows:

   The Configuration Management Process is a process of applying administrative and technical procedures throughout the software life cycle to: *identify and define* software items in a system; control modifications and releases of the items; record and report the status of the items and modification requests; ensure the completeness, consistency, and correctness of the items, and control storage, handling, and delivery of the items.

10. Clause 6.3.4.1. This clause should read as follows:

    Additional quality management activities shall be assured in accordance with the clauses of ISO 9001.

Annex E *(Continued)*

11. Clause 6.5.2. This clause depends heavily on testing (real-time executions) for validation. To allow flexibility, the following note should be added at the end of clause 6.5.2.5:

    NOTE—Other means besides testing (such as, analysis, modeling, simulation, etc.) may be employed for validation.

12. Clause 6.6.3.1.e. This clause should read as follows:

    e) They are ready for the next *planned* activity.

GUIDANCE:

Changes to normative clauses indicated in this annex have been incorporated in the boxed clauses of this guide.

Annex F
(informative)
Use of reusable software products

F.1 Scope

This annex provides guidance on the incorporation of reusable software products.

F.2 Evaluating reusable software products

Examples of candidate criteria that may be used in evaluating reusable software products include, but are not limited to

a) Ability to provide required capabilities and meet required constraints;
b) Ability to provide required safety, security, and privacy protection;
c) Reliability/maturity, as evidenced by established track record;
d) Testability;
e) Interoperability with other system and system-external elements;
f) Distribution issues, including
    1) Restrictions on copying/distributing the software or documentation;
    2) License or other fees applicable to each copy.
g) Maintainability, including
    1) Likelihood the software product will need to be changed;
    2) Feasibility of accomplishing that change;
    3) Availability and quality of documentation and source files;
    4) Likelihood that the current version will continue to be maintained by the producer;
    5) Impact on the system if the current version is not maintained;
    6) The acquirer's usage and ownership rights to the software product;
    7) Warranties available.
h) Short-term and long-term cost impacts of using the software product;
i) Technical, cost, and schedule risks and trade-offs in using the software product.

F.3 Interpreting IEEE/EIA 12207.0's activities for reusable software products

The following rule can be applied in interpreting IEEE/EIA 12207.0: Any requirement that calls for development of a software product may be met by a reusable software product that fulfills the requirement and meets the established criteria. The reusable software product may be used as is or modified and may be used to satisfy part or all of the requirement. For example, a requirement may be met by using an existing plan, specification, or design.

Annex G
(informative)
Candidate joint management reviews

G.1 Scope

This annex describes a candidate set of joint management reviews that might be held during a software project. Additional information on joint reviews may be found in IEEE Std 1028.

G.2 Assumptions

This annex makes the following assumptions:

a) The acquirer has reviewed the subject products in advance, and one or more joint technical reviews have been held to resolve issues, leaving the joint management review as a forum to resolve open issues and reach agreement as to the acceptability of each product;

b) Any of the reviews may be conducted incrementally, dealing at each review with a subset of the listed items or a subset of the system or software items being reviewed.

G.3 Candidate reviews

Given below is a set of candidate joint management reviews that might be held during a software project. There is no intent to require these reviews or to preclude alternatives or combinations of these reviews. Software products are expressed in lower case letters to emphasize that they are generic products, not necessarily in the form of hard-copy documents.

G.3.1 Software plan reviews

These reviews are held to resolve open issues regarding one or more of the following:

a) Various software plans;
b) The software test plan;
c) Development, test, operational, maintenance environment and tools.

G.3.2 Operational concept reviews

These reviews are held to resolve open issues regarding the operational concept for a software system.

G.3.3 System/subsystem requirements reviews

These reviews are held to resolve open issues regarding the specified requirements for a software system or subsystem.

G.3.4 System/subsystem design reviews

These reviews are held to resolve open issues regarding one or more of the following:

a) The system-wide or subsystem-wide design decisions;
b) The architectural design of a software system or subsystem.

G.3.5 Software requirements reviews

These reviews are held to resolve open issues regarding the specified requirements for a software item.

G.3.6 Software design reviews

These reviews are held to resolve open issues regarding one or more of the following:

a) The software-wide design decisions;
b) The architectural design of a software item;
c) The detailed design of a software item or portion thereof (such as a database).

G.3.7 Test readiness reviews

These reviews are held to resolve open issues regarding one or more of the following:

a) The status of the software test environment;
b) The test cases and test procedures to be used for software item qualification testing or system qualification testing;
c) The status of the software to be tested.

G.3.8 Test results reviews

These reviews are held to resolve open issues regarding the results of software item qualification testing or system qualification testing.

G.3.9 Software usability reviews

These reviews are held to resolve open issues regarding one or more of the following:

a) The readiness of the software for installation at user sites;
b) Status of training, including “training software products,” if applicable;
c) The user and operator manuals;
c) The software version descriptions;
e) The status of installation preparations and activities.

G.3.10 Software maintenance reviews

These reviews are held to resolve open issues regarding one or more of the following:

a) The readiness of the software for transition to the maintenance organization;
b) The software product specifications;
c) The software maintenance manuals;
d) The software version descriptions;
e) The status of transition preparations and activities, including transition of the software engineering environment, if applicable.

G.3.11 Critical requirement reviews

These reviews are held to resolve open issues regarding the handling of critical requirements, such as those for safety, security, and privacy protection.

Annex H
(informative)
Software measurement categories

H.1 Scope

This annex identifies the key software measurement categories applicable to software projects. These categories are defined based on the software issues they address. An issue is defined as a current or potential problem area that might impact the achievement of project objectives with respect to cost, schedule, functionality, quality, or performance. Issues include risks, problems, and uncertainties. Software measures are selected based on the prioritized information needs related to the defined project issues. The selected measures are then defined specifically within the context of the life cycle processes, tasks, and activities inherent to the project. Additional information on software measurement categories is available in IEEE Std 982.1, IEEE Std 1044, IEEE Std 1045, IEEE Std 1061, and in *Practical Software Measurement: A Guide to Objective Program Insight*.

H.2 Candidate measurement categories

The specific software issues and associated software measures are uniquely defined for each project. The issues and overall priority of the software measurement information requirements change over the project life cycle. As a result, different measures are used at different times within the project and are dependent upon the life cycle processes, tasks, and activities taking place. Most software issues can be grouped into one or more of the issue categories listed below. These issue categories provide a starting point for selecting the appropriate software measures. The issue categories may be adapted for each project. “User Satisfaction,” for example, could be defined as an issue category for a “shrink-wrapped commercial software project.” Each issue category is associated with specific types of software measures. Both the issue categories and the types of measures are related and should be implemented using an integrated approach.

H.2.1 Schedule and progress

This category addresses specific software issues regarding completion of program milestones, significant project and software events, and individual software work items.

H.2.2 Resources and cost

This category addresses specific software issues regarding the balance between work to be performed and personnel and financial resources allocated to the project.

H.2.3 Growth and stability

This category addresses specific software issues regarding the stability of the software requirements, functionality, or capability and regarding the volume of software delivered to provide required capability.

H.2.4 Product quality

This category addresses specific software issues regarding the ability of the delivered product to support the user’s needs and requirements. It also addresses discovered problems and errors that result in the need for rework.

H.2.5 Development performance

This category addresses specific software issues regarding the supplier productivity and overall software process capabilities relative to project needs.

H.2.6 Technical adequacy

This category addresses project unique software issues related to the use of new or diverse software technologies. It relates to the viability of the proposed technical approach with respect to project returns (for example, large planned resource savings from the reuse of existing software components).

Annex I
(informative)
Guidance on development strategies and build planning

I.1 Scope

This annex describes three candidate development strategies and shows how IEEE/EIA 12207.0 may be applied under each of these strategies and on a project involving reengineering.

I.2 Candidate development strategies

There are many different development strategies that can be applied to software projects. Three of these strategies are summarized below and in figure I.1.

a) Once-through. The “once-through” strategy, also called “waterfall,” consists of performing the development process a single time. Simplistically: determine user needs, define requirements, design the system, implement the system, test, fix, and deliver.

b) Incremental. The “incremental” strategy determines user needs and defines the system requirements, then performs the rest of the development in a sequence of builds. The first build incorporates part of the planned capabilities, the next build adds more capabilities, and so on, until the system is complete.

c) Evolutionary. The “evolutionary” strategy also develops a system in builds but differs from the incremental strategy in acknowledging that the user need is not fully understood and all requirements cannot be defined up front. In this strategy, user needs and system requirements are partially defined up front, then are refined in each succeeding build.

Development strategy	Define all requirements first?	Multiple development cycles?	Distribute interim software?
Once-Through (Waterfall)	Yes	No	No
Incremental (Preplanned Product Improvement)	Yes	Yes	Maybe
Evolutionary	No	Yes	Yes

Figure I.1—Key features of three development strategies

I.3 Selecting an appropriate development strategy

The development strategy should be agreed on by the acquirer and the supplier. The acquirer should carefully state any program-specific requirements that may affect the selection of a development strategy (e.g., funding profile limitations, early prototype demonstrations, evolving operational requirements). The supplier should propose, subject to acquirer approval, the development strategy that most effectively satisfies the program requirements. A specific strategy should be required by the acquirer only when the acquirer has a complete understanding of the programmatic and technical domains and determines that no other approach will be entertained.

Figure I.2 illustrates a risk analysis approach for selecting an appropriate strategy. The approach consists of listing risk items (negatives) and opportunity items (positives) for each strategy; assigning each item a risk or opportunity level of High, Medium, or Low; and making a decision on which strategy to use based on a trade-off among the risks and opportunities. The fill-ins shown are sample considerations only. An actual analysis may use others. The “DECISION” entry on the bottom line shows which strategy was selected.

I.4 Relationship of IEEE/EIA 12207.0 to development strategies

The development strategy usually applies to the overall system. The software within the system may be acquired under the same strategy or under a different one, such as requiring that all software be finalized in the first build of the system. Figures I.3, I.4, and I.5 show how IEEE/EIA 12207.0 might be applied under each of the development strategies identified in I.2. Figure I.6 shows how IEEE/EIA 12207.0 might be applied on a reengineering project. All four figures are, by necessity, simplified. For example, they show the standard’s activities in sequence when they might actually be ongoing, overlapping, or iterative, and they show each software product as a single entity, without depicting early drafts or updates.

I.5 Planning software builds

Planning the software builds on a project may be accomplished in several ways. The acquirer might, for example, select an overall development strategy, leaving it to the supplier to lay out the software builds. Alternatively, the acquirer might lay out the software builds as part of the contract. The approach selected will be project-dependent. The subclauses below provide guidelines for planning the builds without attempting to identify who performs each of the activities.

I.5.1 Identifying builds and their objectives

The first step in software build planning is to lay out a series of one or more builds and to identify the objectives of each build.

I.5.2 Identifying the activities to be performed in each build

The next step in build planning is identifying which activities apply in each build and determining the extent to which they apply. Some activities may not apply at all in a given build, some may apply identically in all builds, and some may apply differently in different builds. The following guidelines apply:

a) Different decisions may apply to different types of software on a project.

b) If early builds are devoted to experimentation, developing “throw-away” software to arrive at a system concept or system requirements, it may be appropriate to forgo certain formalities, such as coding standards, that may be imposed later on the “real” software. If the early software is to be used later, such formalities may be appropriate from the start. These decisions are project-dependent.

I.5.3 Recording build planning decisions

Build planning decisions made by the acquirer before the project begins are specified in the contract. Build planning proposed by the supplier may be communicated via feedback on draft requests for proposal, proposals, the software plan, joint reviews during the project, or by other means of communication. Refinements to the build planning decisions may be ongoing as the project proceeds. Those decisions involving contractual changes are handled accordingly.

Once-Through		Incremental		Evolutionary	
Risk Item (Reasons against this strategy)	Risk Level	Risk Item (Reasons against this strategy)	Risk Level	Risk Item (Reasons against this strategy)	Risk Level
- Requirements are not well understood	H	- Requirements are not well understood	H		
- System too large to do all at once	M	- User prefers all capabilities at first delivery	M	- User prefers all capabilities at first delivery	M
- Rapid changes in technology anticipated—may change the requirements	H	- Rapid changes in technology are expected—may change the requirements	H		
- Limited staff or budget available now	M				
Opportunity Item (Reasons to use this strategy)	Opp. Level	Opportunity Item (Reasons to use this strategy)	Opp. Level	Opportunity Item (Reasons to use this strategy)	Opp. Level
- User prefers all capabilities at first delivery	M	- Early capability is needed	H	- Early capability is needed	H
- User prefers to phase out old system all at once	L	- System breaks naturally into increments	M	- System breaks naturally into increments	M
		- Funding/staffing will be incremental	H	- Funding/staffing will be incremental	H
				- User feedback and monitoring of technology changes is needed to understand full requirements	H
				DECISION: USE THIS STRATEGY	

Figure I.2—Sample risk analysis for determining the appropriate development strategy

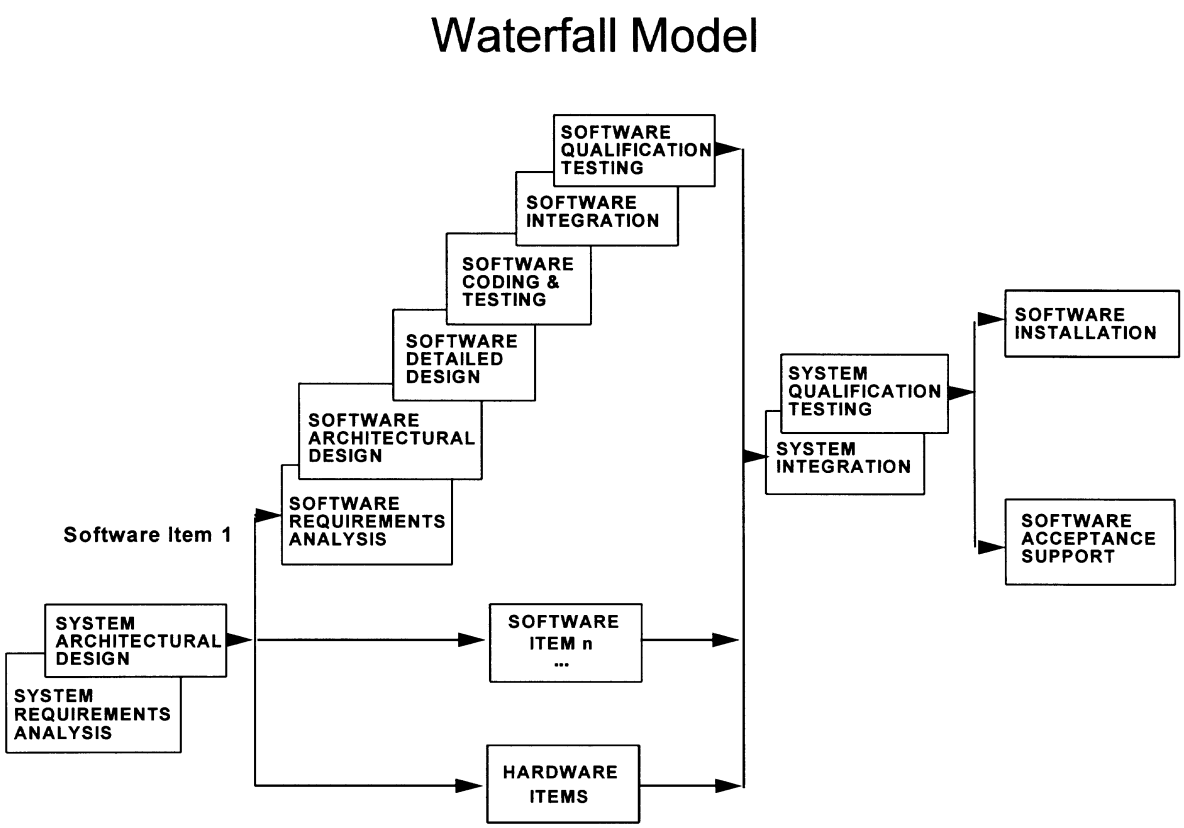

Figure I.3— One possible way of applying IEEE/EIA 12207.0 to the Waterfall development strategy

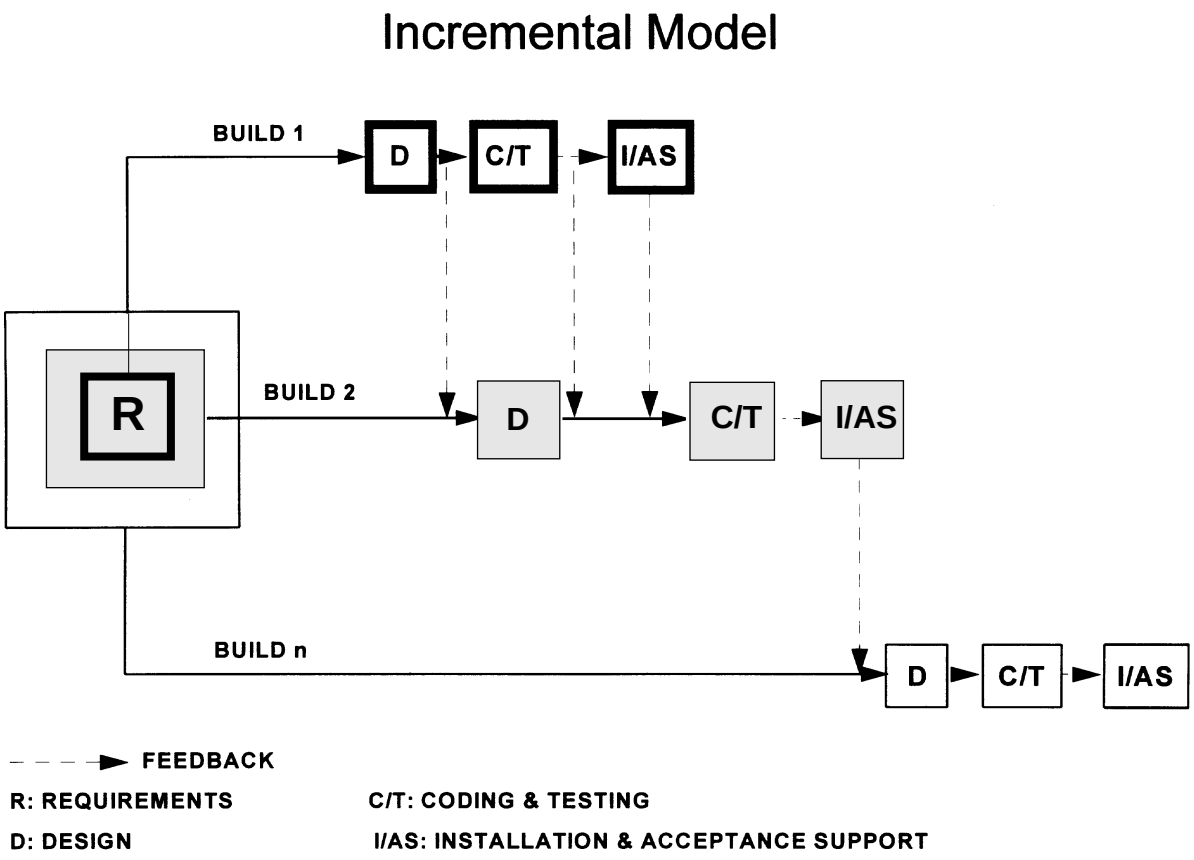

Figure I.4— One possible way of applying IEEE/EIA 12207.0 to the Incremental development strategy

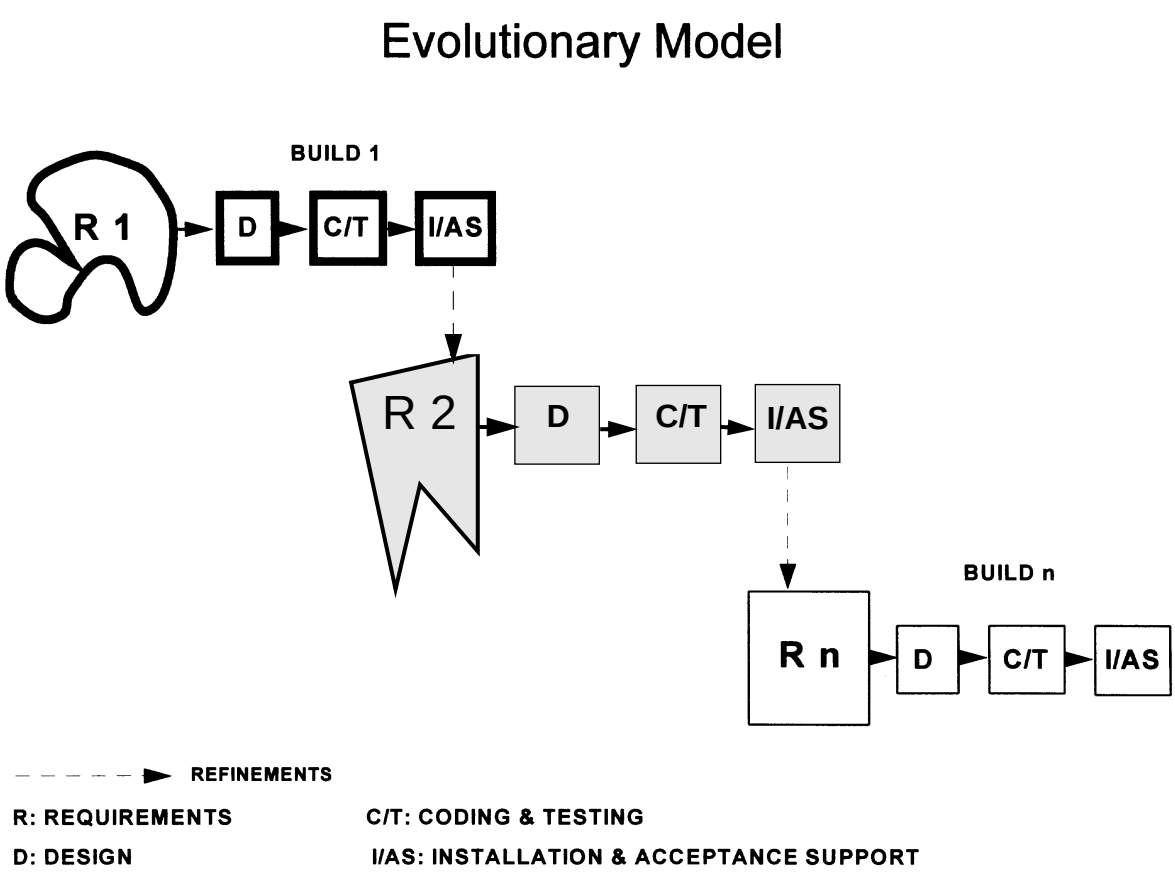

Figure I.5— One possible way of applying IEEE/EIA 12207.0 to the Evolutionary development strategy

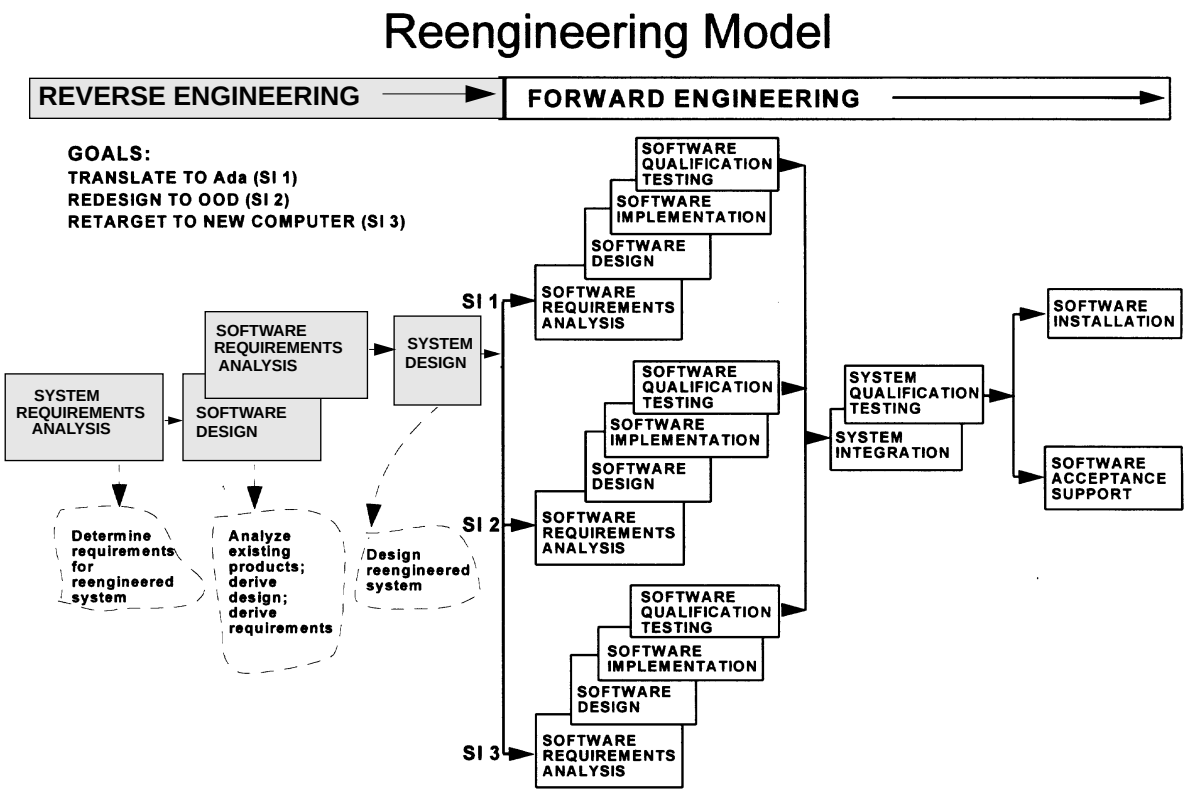

Figure I.6— One possible way of applying IEEE/EIA 12207.0 to a reengineering project

I.5.4 Scheduling the selected activities in each build

Another important step in build planning is scheduling the activities in each build. The acquirer may set forth general milestones and have the supplier provide specifics or may provide specific schedules. The following guidelines apply:

a) A common mistake is to treat all software items as though they are required to be developed in "lock-step," reaching key milestones at the same time. Allowing software items to be on different schedules can result in more optimum development.

b) A similar mistake is to treat software units as though they are required to be developed in "lock-step," all designed by a certain date, implemented by a certain date, etc. Flexibility in the scheduling of software units can also be effective.

c) The activities in the standard need not be performed sequentially. Several may be taking place at one time, and an activity may be performed continually or intermittently throughout a build or over multiple builds. The activities in each build are to be laid out in the manner that best suits the work to be done.

Annex J
(informative)
Category and priority classifications for problem reporting

J.1 Scope

IEEE Std 1044 may be useful in developing classifications for problem reporting.

J.2 Classification by category

The developer should assign each problem in software products to one or more of the categories in figure J.1.

Category		Applies to problems in
a.	Plans	One of the plans developed for the project
b.	Concept	The operational concept
c.	Requirements	The system or software requirements
d.	Design	The design of the system or software
e.	Code	The software code
f.	Database/data file	A database or data file
g.	Test information	Test plans, test descriptions, or test reports
h.	Manuals	The user, operator, or maintenance manuals
i.	Other	Other software products

Figure J.1—Categories to be used for classifying problems in software products

J.3 Classification by priority

The developer should assign each problem in software products or activities to one of the priorities in figure J.2.

Priority	Applies if a problem could
1	a) Prevent the accomplishment of an essential capability b) Jeopardize safety, security, or other requirement designated “critical”
2	a) Adversely affect the accomplishment of an essential capability and no work-around solution is known b) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, and no work-around solution is known
3	a) Adversely affect the accomplishment of an essential capability but a work-around solution is known b) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, but a work-around solution is known
4	a) Result in user/operator inconvenience or annoyance but does not affect a required operational or mission-essential capability b) Result in inconvenience or annoyance for development or maintenance personnel but does not prevent the accomplishment of the responsibilities of those personnel
5	Any other effect

Figure J.2—Priorities to be used for classifying problems

Annex K
(informative)
Software product evaluations

K.1 Scope

Several international standards may be useful in software product evaluation: ISO/IEC 9126, ISO/IEC 12119, and ISO 9127. This annex identifies criteria that may be used for software product evaluations and contains a suggested default set of definitions for the evaluation criteria. When problems are identified, annex J of this guide may be used to establish “error categories” for trend analysis and preventative action.

K.2 Software product evaluations

Evaluations of system-level products are to be interpreted as participation in these evaluations. Some of the criteria are subjective; therefore, there is no requirement to prove that the criteria have been met. The requirement is to perform the evaluations and to identify possible problems for discussion and resolution.

K.3 Criteria definitions

The following subclauses provide definitions for the evaluation criteria that may not be self-explanatory. The criteria are listed in alphabetical order.

K.3.1 Accurately describes (an item)

This criterion, applied to user/operator/programmer instructions and to the “as built” design and version descriptions, means that the instructions or descriptions are correct depictions of the software or other item described.

K.3.2 Adequate test cases, procedures, data, results

Test cases are adequate if they cover all applicable requirements or design decisions and specify the input to be used, the expected results, and the criteria to be used for evaluating those results. Test procedures are adequate if they specify the steps to be followed in carrying out each test case. Test data are adequate if they enable the execution of the planned test cases and test procedures. Test or dry run results are adequate if they describe the results of all test cases and show that all criteria have been met, possibly after revision and retesting.

K.3.3 Consistent with indicated product(s)

This criterion means that (1) no statement or representation in one software product contradicts a statement or representation in the other software products; (2) a given term, acronym, or abbreviation means the same thing in all of the software products; and (3) a given item or concept is referred to by the same name or description in all of the software products.

K.3.4 Covers (a given set of items)

A software product “covers” a given set of items if every item in the set has been dealt with in the software product. For example, a design covers a set of requirements if every requirement has been dealt with in the design; a test plan covers a set of requirements if every requirement is the subject of one or more tests. “Covers” corresponds to the downward traceability (for example, from requirements to design) in the requirements, design, and test planning/descriptions.

K.3.5 Feasible

This criterion means that, based on the knowledge and experience of the evaluator, a given concept, set of requirements, design, test, etc. violates no known principles or lessons learned that would render it impossible to carry out.

K.3.6 Follows software plan

This criterion means that the software product shows evidence of having been developed in accordance with the approach described in the software plan. Examples include following design and coding standards described in the plan. For the software plan itself, this criterion applies to updates to the initial plan.

K.3.7 Internally consistent

This criterion means that (1) no two statements or representations in a software product contradict one another; (2) a given term, acronym, or abbreviation means the same thing throughout the software product; and (3) a given item or concept is referred to by the same name or description throughout the software product.

K.3.8 Meets delivery requirements, if applicable

This criterion applies if the software product being evaluated is deliverable and has been formatted for delivery at the time of evaluation. It focuses on the format, markings, and other provisions specified in the contract, rather than on content, covered by other criteria.

K.3.9 Presents a sound approach

This criterion means that, based on the knowledge and experience of the evaluator, a given plan represents a reasonable way to carry out the required activities.

K.3.10 Shows evidence that (an item under test) meets its requirements

This criterion means that recorded test results show that the item under test either passed all tests the first time or was revised and retested until the tests were passed.

K.3.11 Testable

A requirement or set of requirements is considered to be testable if an objective and feasible test can be designed to determine whether each requirement has been met.

K.3.12 Understandable

This criterion means “understandable by the intended audience.” For example, software products intended for programmer-to-programmer communication need not be understandable by nonprogrammers. A product that correctly identifies its audience and is considered understandable to that audience meets this criterion.

Annex L
(informative)
Risk management

L.1 Scope

Risk management is part of the overall management process related to potential technical, cost, and schedule risks. Risk management is a tool for problem prevention: identifying the things that could go wrong, assessing their impact, and determining which potential problems need to be avoided. Risk management is an activity that should be performed by all levels of the project to ensure adequate coverage of all potential problem areas. Open communication is required to provide all project personnel with the freedom to identify issues without negative consequences to themselves. Joint management of risks between acquirer and supplier, with support from developer/maintainer/operator, is necessary to enable identification of the most important risks to the program and to support efficient allocation of mitigation resources. Risk management may make use of the results of other supporting processes, such as problem resolution, quality assurance, and joint reviews.

L.2 Risk planning

A risk management plan should be developed. This plan should describe risk management activities, procedures and schedules for performing these activities, documentation and reporting requirements, organizations and personnel responsible for performing specific activities, and requirements for communicating risks and risk status with other organizations (e.g., acquirer, supplier, subcontractor).

NOTE—This plan may be part of the project management plan.

L.3 Risk identification

Methods should be established for the identification of the program's risks, including a statement of risk and a unique identifier, date of identification, and additional context information necessary to fully understand the risk.

L.4 Risk analysis

Risks should be classified or grouped into related sets for optimal effectiveness of management and mitigation. Classification should be done using a consistent classification scheme and basis. Risk attributes of probability, impact, and time frame of occurrence should be evaluated. Qualitative and quantitative methods should be used as appropriate to the nature of the risk. Risks should be prioritized to determine their relative importance to the program and to provide a basis for effective use of mitigation resources.

L.5 Risk mitigation

The responsibility for managing each risk should be assigned to an appropriate person. The responsibility should include development of mitigation plans, tracking risk and mitigation plan progress, and reporting risk status. Risks that are best managed outside of the organization should be transferred (e.g., from a supplier to an acquirer). The risk classification scheme should include risk status. Any scheme deemed appropriate may be used, including the following scheme, which assigns a risk status of closed, monitored, or mitigated:

a) Closed indicates no further action should be taken regarding the risk. Closed risks should include documented rationale for closure.

b) Monitored indicates the risk should be actively tracked with defined measures, milestones, or metrics.

c) Mitigated indicates the risk should have documented mitigation plans specifying the goal of mitigation, actions to be taken, responsibility for actions, metrics for monitoring the risk, contingency triggers and plans (if needed), schedules, and costs.

L.6 Risk tracking and control

Risks and mitigation plans should be tracked for significant changes in attributes, predefined triggers, thresholds, events, or mitigation plan progress or failure. Status reports required for the project should include risk management information.

Upon review, risks should be closed when their probability of impact has reached a sufficiently low level or their mitigation plans have succeeded. Failing or unsuccessful mitigation plans should be revised. Contingency actions specified in mitigation plans should be implemented if contingency triggers occur. Joint resolution agreements should be required for controlling actions where the risks are jointly visible or jointly controlled.

Annex M
(informative)
Life cycle processes references

M.1 Scope

The following table contains the ISO/IEC 12207 (IEEE/EIA 12207.0) life cycle process clause numbers and a list of available standards/guides/technical reports published by IEEE and ISO/IEC JTC1/SC7 that may be used as references when working with the life cycle processes in IEEE/EIA 12207.0.

IEEE/EIA 12207.0 process	Reference
5.1 Acquisition process	IEEE 982.1, 1062, 1228 RTCA DO-178B
5.2 Supply process	
5.3 Development process	IEEE 829, 830, 1008, 1016, 1016.1, 1028, 1074, 1074.1, 1228 ISO/IEC 9126
5.4 Operation process	
5.5 Maintenance process	IEEE 1219
6.1 Documentation process	ISO 9127, ISO/IEC TR 9294
6.2 Configuration management process	IEEE 828, 1012, 1042, 1059 ISO 10007
6.3 Quality assurance process	IEEE 730, 730.1, 1061, 1298 ISO 9001, 9000-3, 10005
6.4 Verification process	IEEE 1012, 1028, 1059
6.5 Validation process	IEEE 1012, 1028, 1059
6.6 Joint review process	IEEE 1028
6.7 Audit process	IEEE 1028
6.8 Problem resolution process	IEEE 1044, 1044.1
7.1 Management process	IEEE 1045, 1058.1
7.2 Infrastructure process	IEEE 1209, 1348, 1420.1 ISO/IEC 14102
7.3 Improvement process	—
7.4 Training process	—

NOTE—EIA/IEEE J-STD-016:1995 may be a useful reference, particularly for existing systems.

M.2 References

The titles of the standards/guides/reports cited in this guide are listed below. When the following standards are superseded by an approved revision, the revision shall apply.

EIA/IEEE J-STD-016:1995, *Standard for Information Technology—Software Life Cycle Processes—Software Development: Acquirer-Supplier Agreement*.[2]

IEEE Std 730-1989, *IEEE Standard for Software Quality Assurance Plans* (ANSI).[3]

[2] EIA/IEEE J-STD-016: 1995 is available from Global Engineering, 1990 M Street NW, Suite 400, Washington, DC, 20036, USA. It is also available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

[3] IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

IEEE Std 730.1-1995, *IEEE Guide for Software Quality Assurance Planning* (ANSI).

IEEE Std 828-1990, *IEEE Standard for Software Configuration Management Plans* (ANSI).

IEEE Std 829-1983 (Reaff 1991), *IEEE Standard for Software Test Documentation* (ANSI).

IEEE Std 830-1993*, IEEE Recommended Practice for Software Requirements Specifications* (ANSI).

IEEE Std 982.1-1988, *IEEE Standard Dictionary of Measures to Produce Reliable Software* (ANSI).

IEEE Std 1008-1987, *IEEE Standard for Software Unit Testing* (ANSI).

IEEE Std 1012-1986 (Reaff 1992), *IEEE Standard for Software Verification and Validation Plans* (ANSI).

IEEE Std 1016-1987 (Reaff 1993), *IEEE Recommended Practice for Software Design Descriptions* (ANSI).

IEEE Std 1016.1-1993, *IEEE Guide to Software Design Descriptions* (ANSI).

IEEE Std 1028-1997, *IEEE Standard for Software Reviews* (ANSI).

IEEE Std 1042-1987 (Reaff 1993), *IEEE Guide to Software Configuration Management* (ANSI).

IEEE Std 1044-1993, *IEEE Standard Classification for Software Anomalies* (ANSI).

IEEE Std 1044.1-1995, *IEEE Guide to Classification for Software Anomalies* (ANSI).

IEEE Std 1045-1992, *IEEE Standard for Software Productivity Metrics* (ANSI).

IEEE Std 1058.1-1987 (Reaff 1993), *IEEE Standard for Software Project Management Plans* (ANSI).

IEEE Std 1059-1993, *IEEE Guide for Software Verification and Validation Plans* (ANSI).

IEEE Std 1061-1992, *IEEE Standard for a Software Quality Metrics Methodology* (ANSI).

IEEE Std 1062-1993, *IEEE Recommended Practice for Software Acquisition* (ANSI).

IEEE Std 1074-1995, *IEEE Standard for Developing Software Life Cycle Processes* (ANSI).

IEEE Std 1074.1, *IEEE Guide for Developing Software Life Cycle Processes* (ANSI).

IEEE Std 1209-1992, *IEEE Recommended Practice for the Evaluation and Selection of CASE Tools* (ANSI).

IEEE Std 1219-1992, *IEEE Standard for Software Maintenance* (ANSI).

IEEE Std 1228-1994, *IEEE Standard for Software Safety Plans* (ANSI).

IEEE Std 1298-1992 (AS 3563.1-1991), *Software Quality Management System, Part 1: Requirements* (ANSI).

IEEE Std 1348-1995, *IEEE Recommended Practice for the Adoption of Computer-Aided Software Engineering (CASE) Tools* (ANSI).

IEEE Std 1420.1-1995, *IEEE Standard for Information Technology—Software Reuse—Data Model for Reuse Library Interoperability: Basic Interoperability Data Model (BIDM)* (ANSI).

ISO 9000-3:1997, *Quality management and quality assurance standards—Part 3: Guidelines for the application of ISO 9001:1994 to the development, supply, installation and maintenance of computer software*.[4]

ISO 9001:1994*, Quality systems—Model for quality assurance in design, development, production, installation and servicing*.

ISO 9127:1988, *Information processing systems—User documentation and cover information for consumer software packages*.

ISO 10005:1995, *Quality management—Guidelines for quality plans* (formerly ISO/DIS 9004-5).

ISO 10007:1995, *Quality management—Guidelines for configuration management*.

ISO/IEC 9126:1991, *Information technology—Software product evaluation—Quality characteristics and guidelines for their use*.

ISO/IEC 12119:1994*, Information technology—Software packages—Quality requirements and testing*.

ISO/IEC 14102:1995*, Information technology—Guideline for the evaluation and selection of CASE tools*.

ISO/IEC TR 9294:1990, *Information technology—Guidelines for the management of software documentation*.

Practical Software Measurement: A Guide to Objective Program Insight.[5]

RTCA DO-178B, *Software Considerations in Airborne Systems and Equipment Certification*.[6]

[4] ISO and ISO/IEC publications are available from the ISO Central Secretariat, Case Postale 56, 1 rue de Varembé, CH-1211, Genève 20, Switzerland/Suisse. ISO publications are also available in the United States from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

[5] This document was published by the U.S. Navy, Washington, DC, 1996.

[6] RTCA publications are available from RTCA, 1140 Connecticut Avenue NW, Suite 1020, Washington, DC 20036, USA.